VYSOKÁ ŠKOLA BÁŇSKÁ – TECHNICKÁ UNIVERZITA OSTRAVA
EKONOMICKÁ FAKULTA

KATEDRA APLIKOVANÉ INFORMATIKY

Mobilní aplikace na platformě Android zobrazující školní rozvrhy

Mobile application on the Android platform displaying school schedules

Student: Bc. Michal Kocián

Vedoucí diplomové práce: Ing. Vítězslav Novák, Ph.D.

Ostrava 2014

VŠB - Technická univerzita Ostrava
Ekonomická fakulta
Katedra aplikované informatiky

Zadání diplomové práce

Student: **Bc. Michal Kocián**

Studijní program: N6209 Systémové inženýrství a informatika

Studijní obor: 1802T001 Aplikovaná informatika

Téma: Mobilní aplikace na platformě Android zobrazující školní rozvrhy
Mobile Application on the Android Platform Displaying School Schedules

Zásady pro vypracování:

1. Úvod
2. Teoretická východiska návrhu Android aplikace
3. Návrh aplikace školního rozvrhu a podpůrné webové aplikace
4. Implementace mobilní aplikace a datového serveru
5. Závěr

Seznam použité literatury
Seznam zkratek
Prohlášení o využití výsledků diplomové práce
Seznam příloh
Přílohy

Seznam doporučené odborné literatury:

MEIER, Reto. *Professional Android 4 application development.* 2nd ed. Indianapolis: John Wiley, 2012. ISBN 978-111-8262-153.
SCHWALBE, Kathy. *Řízení projektů v IT: kompletní průvodce.* Přeložila Hana KREJČÍ. Brno: Computer Press, 2011. ISBN 978-80-251-2882-4.
FOWLER, Martin. *UML distilled: a brief guide to the standard object modeling language.* 3rd ed. Boston: Addison-Wesley, 2004. ISBN 978-0-321-19368-1.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Vítězslav Novák, Ph.D.**

Datum zadání: 22.11.2013
Datum odevzdání: 25.04.2014

Ing. Petr Rozehnal, Ph.D.
vedoucí katedry

prof. Dr. Ing. Dana Dluhošová
děkanka fakulty

Prohlašuji, že jsem celou práci, včetně všech příloh, vypracoval(a) samostatně.

……

Bc. Michal Kocián

V Ostravě dne 25.4.2014

Obsah

1 Úvod

Mobilní telefony a tablety se systémem Android jsou k dispozici poměrně krátkou dobu, přesto však zažívají velký rozmach. Z vedlejšího projektu firmy Google Inc. se stal produkt, který znají lidé na celém světě.

Studenti si na začátku školního semestru ještě nepamatují přesnou podobu nových rozvrhů, často je tak vyhledávají přes mobilní telefony na webových stránkách rozvrhu školy, jenže tato stránka není nijak optimalizovaná pro malá dotyková zařízení. Není až tak snadné rychle zjistit potřebné informace pokud není k dispozici počítač, případně Internet na mobilním zařízení.

Stálo za úvahu, jak by se tento rozvrh jednoduše přenesl na mobilní telefony. Dá se použít vytváření záznamů v kalendáři, instalovat různé aplikace, ale zadávání předmětu je vždy komplikované a časově náročné.

Cílem této diplomové práce je navrhnout a vytvořit Android aplikaci, která zprostředkuje zobrazení školních rozvrhů pro studenty a učitele. Zaměřuje se na jednoduché a přehledné podání informací o vyučovacích hodinách. Bude tak zpracována užitečná pomůcka, kterou může mít každý při ruce k nahlédnutí.

Na počátku bylo nejkomplikovanější samotné získání potřebných dat. Ukázalo se, že nejjednodušší cestou je napsání jednoduchého skriptu, který načte všechny jednotlivé stránky studijních skupin a postupně vyparsuje všechna potřebná data. Prvotní aplikaci vidělo několik lidí a setkala se s poměrně velkým ohlasem, což bylo motivací věnovat se tomuto tématu více a následně tuto aplikaci rozvinout do diplomové práce.

V kapitole číslo 2 se zabýváme teoretickými východisky vývoje Android aplikací. Začneme shrnutím historie platformy a objasníme nejdůležitější prvky mobilních technologií. Vysvětlíme si na co si dát pozor.

Kapitola číslo 3 se zabývá návrhem aplikace pro zobrazení rozvrhu školy a podpůrné webové aplikace, která poskytuje data. Je zde rozebráno, jak jsou prvky navrženy, aby fungovaly jako jeden velký celek.

V kapitole číslo 4 si ukážeme jak se vytvářela Android aplikace a její podpůrná webová služba. Představíme si důležité nástroje a ukážeme jejich přednosti. Předvedeme, jak probíhá nasazení kompletní aplikace do obchodu Google Play.

2 Teoretická východiska návrhu Android aplikace

2.1 Historie platformy Android

V počátcích byl systém Android vyvíjen firmou Android Inc., kterou vedl Andy Rubin. Ten po odkoupení v roce 2005 společností Google Inc. měl i nadále, jako hlavní manažer projektu, na starost vývoj systému až do roku 2013, kdy z této vedoucí pozice odstoupil ve prospěch jiného projektu téže firmy. (Tech2crack, 2013)

Dne 5. listopadu 2007 bylo založeno seskupení výrobců mobilních telefonů s názvem Open Handset Aliance. Jeho cílem bylo vyvinout standardní otevřený systém pro mobilní zařízení. Dnešními členy konsorcia jsou společnosti Google, Intel, Samsung, HTC, LG, NVIDIA, Qualcomm, Sony, Dell, Telefónica, T-Mobile, eBay a dalších přibližně 70 společností z celého světa. (Svět Androida, 2013)

V tento den společnost Google také oznámila první verzi mobilního operačního systému Android. Eric Schmidt prohlásil, že má s tímto systémem větší plány než vydat jeden telefon, cílem totiž bylo vytvořit platformu, kterou budou výrobci nasazovat do tisíců různých telefonů běžících na různých hardwarových komponentách. Trvalo téměř celý rok, než byl vydán první telefon s Androidem. Jeho výrobcem byla společnost HTC a ve Spojených Státech byl uveden na trh v říjnu roku 2008. Od té doby jde vývoj rychle dopředu a oblíbenost roste, získává si stále větší popularitu mezi koncovými uživateli.

2.1.1 Android Open Source Project (AOSP)

Android Open Source Project je iniciativa vytvořená pro vedení vývoje mobilní platformy Android. Platforma se skládá z operačního systému, middleware a základních podpůrných aplikací. Nabízí řešení pro širokou škálu mobilních zařízení. Projekt je pod vedením společnosti Google. Zdrojové kódy jsou zde k dispozici na stránkách https://source.android.com/ všem zájemcům, firmám i jednotlivcům, kteří se chtějí dozvědět podrobnější informace o platformě. (Android developers, 2014)

2.1.2 Oblíbenost Androidu roste

Každým rokem se prodá více mobilních telefonů a tabletů se systémem Android, může za to především cenová dostupnost přístrojů oproti jiným platformám a velký počet dostupných

aplikací. Díky své flexibilitě pohání různé typy zařízení od telefonů, přes tablety, notebooky, až po hodinky a brýle.

V současné době je 80% nově vyrobených telefonů poháněno systémem Android. (IDC, 2014)

Obrázek 2.1 Dodávky chytrých telefonů

Top Five Smartphone Operating Systems, Shipments, and Market Share, 4Q 2013 (Units in Millions)					
Operating System	**4Q13 Shipment Volumes**	**4Q13 Market Share**	**4Q12 Shipment Volumes**	**4Q12 Market Share**	**Year-Over-Year Change**
Android	226.1	78.1%	161.1	70.3%	40.3%
iOS	51.0	17.6%	47.8	20.9%	6.7%
Windows Phone	8.8	3.0%	6.0	2.6%	46.7%
BlackBerry	1.7	0.6%	7.4	3.2%	-77.0%
Others	2.0	0.7%	6.7.	2.9%	-70.1%
Total	**289.6**	**100.0%**	**229.0**	**100.0%**	**26.5%**

Zdroj: IDC, 2014

2.1.3 Není Android jako Android

Koncový uživatel zařízení může být často zmaten z toho, jak vypadá uživatelské prostředí jeho zařízení a to především proto, že většina z výrobců mobilních telefonů si jej přizpůsobuje podle svých představ. Jak moc se ono vylepšování výrobcům daří, se dá jen těžko objektivně hodnotit, nicméně nic by se nemělo přehánět.

Některé společnosti, jako Amazon a Nokia, využily otevřenosti kódu AOSP a velkou část původního systému přetvořily k obrazu svému. Jejich zařízení obvykle využívají vlastních obchodů s aplikacemi a snižují tím příjmy společnosti Google Inc. z prodeje.

2.1.4 Kompatibilita aplikace se staršími verzemi systému Android

V průběhu času přináší vývoj svá úskalí díky nově přicházejícím verzím mobilního systému Android. Aby ve všem byl pořádek a vývojáři měli přehled, kdy mohou použít například konkrétní metodu, byly zavedeny úrovně API.

2.1.5 Android API level

Platforma Android zažívá velmi živý vývoj a postupem času je přidáváno velké množství funkcí. Každá nová verze SDK (software development kit) se označuje celým číslem a říká nám, jaké možnosti můžeme v konkrétní verzi platformy využít.

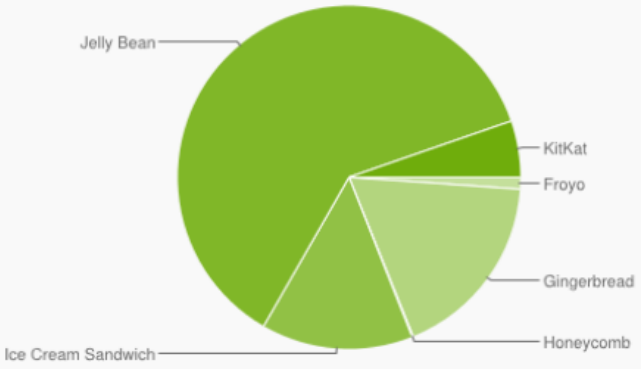
Obsáhlý přehled všech balíčků, které jsou aktuálně k dispozici, nalezneme na stránkách oficiální dokumentace. Pro každého vývojáře je to jedno z nejdůležitějších míst, kde získat základní informace pro vývoj aplikací na platformě Android.

2.1.6 Aktuální zastoupení jednotlivých verzí API

Google každý měsíc vydává statistiky zastoupení jednotlivých verzí API, které vychází z přístupů do obchodu Google Play. Pomocí těchto informací můžeme postupem času zvážit důležitost optimalizací pro starší verze systému. Obvykle platí, že čím starší verze, tím je komplikovanější zajistit zpětnou kompatibilitu všech částí aplikace a vývojáři tak mohou odpadnout spousty starostí s optimalizacemi, které by ocenilo jen velmi malé procento uživatelů.

Obrázek 2.2 Verze platformy Android

Version	Codename	API	Distribution
2.2	Froyo	8	1.1%
2.3.3 - 2.3.7	Gingerbread	10	17.8%
3.2	Honeycomb	13	0.1%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	14.3%
4.1.x	Jelly Bean	16	34.4%
4.2.x		17	18.1%
4.3		18	8.9%
4.4	KitKat	19	5.3%



Data collected during a 7-day period ending on April 1, 2014.
Any versions with less than 0.1% distribution are not shown.

Zdroj: Android developers, 2014

2.1.7 Android Support Library

Tato knihovna zajišťuje zpětnou kompatibilitu napříč staršími verzemi systému Android, takže i když vyvíjíme aplikaci pro nejnovější verzi API (v současné verzi API level 19), můžeme díky této sadě nástrojů využívat některých novinek i pro starší úrovně API. Tímto zajistíme funkčnost jednotlivých komponent aplikace, aniž bychom museli sami náročně vymýšlet způsoby, kterými obejít chybějící komponenty ve starších verzích platformy.

2.2 Hlavní směry vývoje Android aplikací

Aplikace se nejčastěji vytváří pomocí dvou rozličných technologií. Každý způsob má svá úskalí.

2.2.1 Nativní aplikace

Vývoj pomocí standardních komponent Android SDK je výhodný především díky přístupu k funkcím mobilních zařízení a rychlosti výsledné aplikace.

Na druhou stranu však vyžaduje dobré znalosti objektového programování. Programátor však má větší možnosti.

2.2.2 Aplikace jako HTML5

Jednou z možností, jak vytvářet aplikace pro mobilní telefony, je využití HTML5. Psaní HTML5 aplikací je poměrně snadné se naučit a používat, v porovnání s vývojem pomocí nativních technologií. Organizace mohou ušetřit peníze tím, že se vytvoří aplikace, které pracují na všech operačních systémech, místo přepracování aplikace pro každý operační systém.

Možnosti HTML5 nejsou v této práci dále využity, protože mají i poměrně zásadní nevýhody.

- Rychlost takovéto aplikace je na nízké úrovni v porovnání s nativní aplikací. Vývojáři často využívají neefektivní kódovací techniky, výsledný kód je velmi rozsáhlý a tím také hůře udržovatelný.
- Standard HTML5 je poměrně nový a některé jeho komponenty se stále vyvíjí. Je tak pravděpodobné, že na různých platformách, nebo zařízeních, by mohlo dojít k špatné interpretaci kódu některých technologií, jako jsou CSS a JavaScript API.
- Mobilní web aplikace může ke svému běhu potřebovat neustálé datové připojení, což při dnešních cenách není pro každého samozřejmost.
- Tyto aplikace často nedodržují pravidla pro doporučený design a navigaci v aplikaci pro konkrétní platformy.

2.3 Běh Android aplikace

Každá aplikace běží ve vlastním Sandboxu, který izoluje data a spustitelné soubory aplikace od ostatních aplikací v zařízení. Pro své potřeby tak využívá úložiště dat, do kterého nemá

přístup nikdo jiný než samotná aplikace, a tím jsou zabezpečena data, aby neunikaly díky jiným nainstalovaným aplikacím.

2.3.1 Získání dat pro potřebu aplikace

Vnitřní úložiště, do kterého má přístup pouze konkrétní aplikace, která data vlastní.

Externí úložiště, čímž jsou především SD karty v telefonu, které ovšem jsou přístupné pro všechny aplikace ke čtení a zápisu (za předpokladu, že konkrétní aplikace mají povolená práva k přístupu na SD kartu).

Poskytovatelé dat mohou být také samotné aplikace, které implementují rozhraní **Content provider** a zpřístupní tak svá vlastní soukromá data dalším aplikacím. Takto například pracují aplikace jako je seznam kontaktů, který sdílí data například s aplikací pro zasílání zpráv.

Data nemusí být vždy k dispozici lokálně v zařízení, ale mohou pocházet z připojeného příslušenství, nebo webových služeb.

2.3.2 Uživatelská přívětivost

Mobilní aplikace by měly jednoduchou cestou zprostředkovávat informace, které bychom nejraději měli stále po ruce. Častou chybou bývá, že se vývojáři aplikací snaží nabídnout co největší množství funkcí, což může být často pro uživatele matoucí. Uživatelé se obecně neradi učí novým věcem a je velmi pravděpodobné, že pokud narazí na komplikované ovládání, nebudou aplikaci rádi používat a časem ji odinstalují, protože pro ně nebude mít žádný přínos. Abychom tomu předešli, je potřeba, aby vše bylo přehledné, pěkné na pohled a uživatel měl informace dostupné co nejjednodušším způsobem. Platí zde pravidlo, že méně je někdy více a tak je vhodné zaměřit se na kvalitu předávání informací než na kvantitu funkcí, které stejně málo kdo využije.

2.3.3 Dotykové ovládání

Efektivní navigace je jedním ze základních kamenů dobře navržené aplikace. Moderní mobilní zařízení naprostá většina lidí ovládá dotykem prstu a tak je vhodné, aby ovládací prvky měly dostatečně velké rozměry a byly rozpoznatelné na pohled, aby nedocházelo ke zbytečným nežádoucím přehmatům.

Nejčastěji využívaná jsou následující dotyková gesta:

Jednoduché kliknutí je nejčastěji používaným dotykovým gestem, spouští výchozí akci pro daný uživatelský prvek.

Dlouhé stisknutí vyvolává výběrový mód, kdy je možné vybrat jednu, či více položek v seznamu.

Přetažení (swipe) odroluje stávající obsah a zobrazí nové okno. Je to velmi rychlý a efektivní způsob navigace mezi záložkami.

Stisknutím dvěma prsty a tažením od sebe, nebo k sobě (pinch to zoom) se využívá především k přibližování a oddalování obsahu.

Android umožňuje vytváření vlastních dotykových gest a tak není nutné se spoléhat pouze na ta výchozí. Je ovšem na zvážení jestli je v dané situaci vhodné učit uživatele nová gesta, nebo se spolehnout na ty standardní.

2.4 Omezení mobilních zařízení

Mobilní zařízení, ať už se to týká jakékoli mobilní platformy, jsou určeny především pro zobrazení informací, které byly již zpracovány předem. Není vhodné je využívat k zpracování velkých objemů dat, dále uvedeme důvody proč tomu tak je.

2.4.1 Omezený přístup k síti internetu

I když se situace s mobilním internetem stále zlepšuje, především co se cen týče, stále ještě není mobilní internet pro všechny samozřejmostí.

Nedostupnost datového připojení můžeme vyřešit několika způsoby:

- Potřebná data mohou být uložena v lokálním úložišti aplikace do databáze, nebo jako běžné soubory.
- Každý výsledek vyhledávání rozvrhu uložit do cache paměti a při nedostupnosti datového připojení tyto data využít.
- Mohli bychom také jednoduše oznámit, že k akci je potřebné mít dostupný Internet, ale tohle řešení není příliš uživatelsky přívětivé.

2.4.2 Omezená doba provozu na baterii

V dnešní době každý majitel mobilního zařízení je poměrně zásadně omezován výdrží svého přístroje při běhu na baterii. Jelikož každé provedení operace na přístroji je vykoupeno spotřebou energie, je třeba dbát na to, abychom prováděli všechny úkony co možná nejefektivněji.

Jedním z největších konzumentů energie je využívání datových přenosů v mobilní síti, Wi-Fi, či Bluetooth.

2.4.3 Omezený přenos dat

Přenos dat nejenže spotřebovává mnoho energie, ale navíc uživatele může stát dodatečné náklady u mobilních operátorů, proto je třeba navrhnout dobrou strategii jakým způsobem potřebná data získávat a jak často. Jedním ze způsobů je například sledování aktivace Wi-Fi připojení, kdy aplikace spustí aktualizaci až v době, kdy jsme připojeni například do domácí sítě a nehrozí tak překročení datových limitů u mobilních operátorů.

2.5 Využití webových služeb REST

Jedná se o architekturu navrženou pro distribuovaná prostředí, kterou v roce 2000 představil Roy Fielding (jeden ze spoluautorů protokolu HTTP) ve své disertační práci. REST (Representational State Transfer) je, na rozdíl od známějších XML-RPC či SOAP, orientován datově, nikoli procedurálně. Webové služby definují vzdálené procedury a protokol pro jejich volání, REST určuje, jak se přistupuje k datům.

Rozhraní REST je použitelné pro jednotný a snadný přístup ke zdrojům, kterými mohou být data, stejně jako stavy aplikace. Každý datový zdroj má vlastní identifikátor URI a REST definuje čtyři základní metody pro přístup k nim. K přenosu dat je využito datového formátu JSON, který je jazykově nezávislý, má jednoduše čitelnou podobu a především se bude lépe zpracovávat.

2.5.1 Metoda GET

Základní metoda používaná k získávání dat. Je to ekvivalent k obyčejnému požadavku na webovou stránku. Využívá pouze query parametrů k upřesnění dotazu. Na vstupu není žádné tělo zprávy.

```
Příklad:

Požadavek:
GET http://rozvrh.kocian.name/api/v1/semester

Odpověď:
Status: HTTP/1.1 200 OK
Tělo odpovědi:
{
    "semesters": [
        {
            "type": "summer",
            "start": "2014-02-10",
            "end": "2014-07-15"
        }
    ]
}
```

2.5.2 Metoda POST

Tato metoda se používá pro vytváření nových záznamů, kdy ještě není znám identifikátor zdroje. Vstup obvykle obsahuje tělo zprávy, kde jsou doplněny vstupní parametry, které mohou mít i větší velikost než ty obsažené v query řetězci. V této aplikaci je využita pro aktualizaci záznamů v databázi na straně Android aplikace.

2.5.3 Metoda PUT

Úprava stávajících dat se provádí metodou PUT, kdy v těle zprávy zasíláme změněné hodnoty. Identifikátor upravovaného záznamu musí být předem znám.

2.5.4 Metoda DELETE

Jak již název napovídá, slouží metoda k mazání záznamů, kdy data k odstranění jsou určena zadaným identifikátorem ve zprávě.

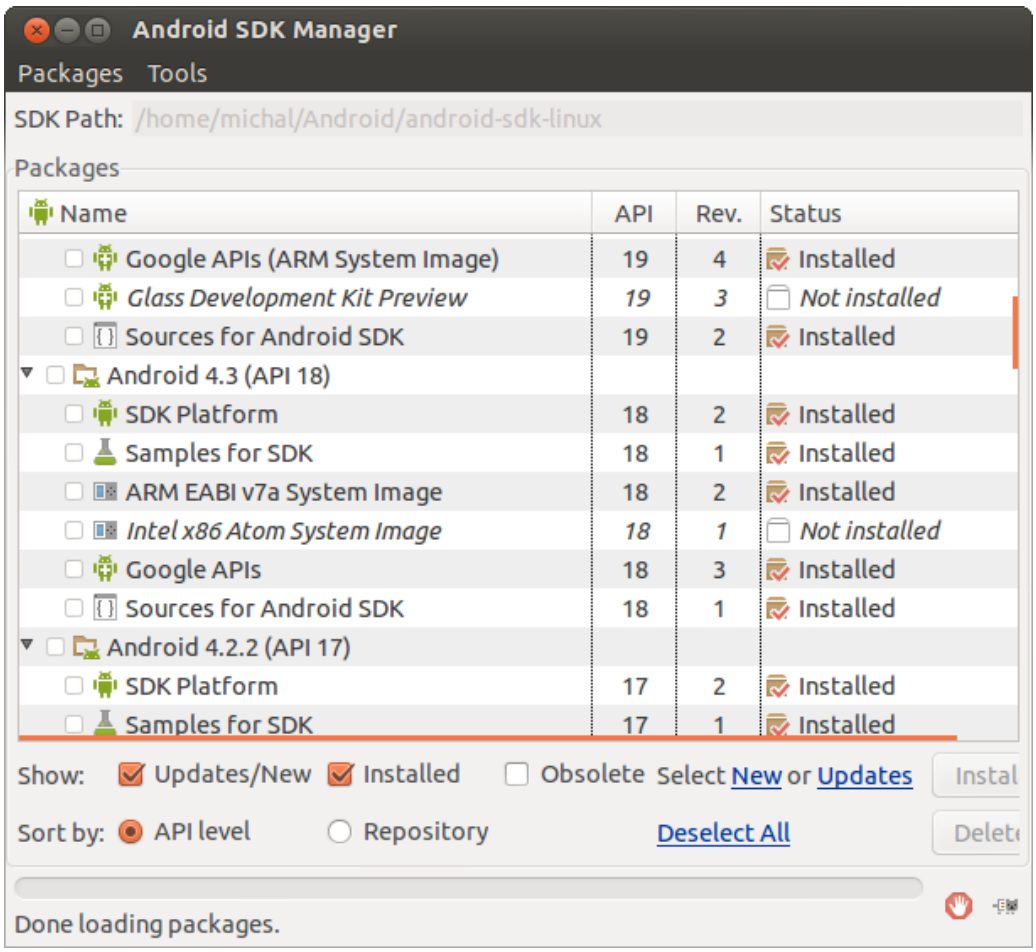
2.6 Nástroje pro vývoj

Vývojář pro svou práci potřebuje nástroje, aby mohl vytvářet a soupštět nové aplikace. K tomu potřebuje stáhnout podpůrné aplikace. Ze začátku si bohatě vystačíme s výchozí nabídkou produktů, ty jsou většinou zdarma. Je tak jen málo překážek, které by nám bránily začít s vývojem.

- Stáhneme Android SDK
- Stáhneme AndroidStudio – je stále ve vývojové verzi

Klíčovým nástrojem pro nás bude nástroj Android SDK Manager, díky němu budeme moct stahovat poporu pro vývoj všech verzí systému,

Obrázek 2.3 Android SDK Manager



Zdroj: Vlastní zpracování

2.7 Testování aplikace

Při vývoji je potřeba spouštět aplikaci a ověřovat její správný chod. Android SDK nám k tomuto dává k dispozici různé podpůrné aplikace pro odladění vytvářeného programového kódu.

K dispozici máme dva způsoby, kde můžeme aplikaci spouštět a to na reálném zařízení, nebo na emulátoru.

2.7.1 Testování na emulátoru

Vývojová platforma Android obsahuje mobilní emulátor zařízení, které běží přímo v počítači. Je tak umožněno vyvíjet a testovat aplikace bez potřeby vlastnit velké množství zařízení s různými parametry, jako jsou především úhlopříčka displeje a verze systému Android. Nevýhodou může být nižší rychlost na slabších počítačích, nebo nemožnost testovat některé funkce jako jsou například gesta více prsty.

Pro potřeby testování si můžeme vytvořit neomezené množství různých emulátorů.

2.7.2 Testování na fyzickém zařízení

Každý mobilní telefon nebo tablet je možno uvést do vývojářského režimu, ve kterém je možné instalovat aplikace přímo z počítače.

Abychom k tomuto účelu mohli zařízení využít, je potřeba upravit nastavení telefonu.

- Připojit zařízení k počítači pomocí kabelu USB.
- Povolit možnost instalace aplikací z neznámých zdrojů.
- Několikrát v nastavení kliknout na název sestavení systému a uvést tak telefon do vývojářského režimu.

2.8 Publikace aplikace na Google Play

Dokončené aplikace je možno nabízet koncovým uživatelům prostřednictvím oficiálního specializovaného obchodu Google Play. Zde máme možnost vložit svou aplikaci a zajistit jí základní propagaci doplněním popisných textů a snímků obrazovky.

Tento obchod není jediný dostupný, jsou zde další, jako například Amazon Appstore a Apple App Store. Aplikace obsažené v těchto zdrojích však neprocházejí tak důkladnou kontrolou na nežádoucí chování kódu a uživatelé tak mohou snadněji přijít k programům obsahujícím malware.

3 Návrh aplikace školního rozvrhu a podpůrné webové aplikace

Kvalitního produktu dosáhneme především tím, že si před implementací ujasníme požadavky na aplikaci a zpracujeme předběžný návrh implementace.

3.1 Analýza současného stavu

V současné době neexistuje žádná mobilní aplikace zaměřující se na poskytování rozvrhů Vysoké školy báňské – Technické univerzity Ostrava.

Studenti začátkem každého semestru řeší, kam si zapsat svůj nový rozvrh hodin. Popíšeme si některé situace, se kterými se můžeme setkat. Většina z následujících řešení naráží na problém průběžné aktualizace informací.

Tisk rozvrhu - vyžaduje dodatečné náklady a pokud se během několika dní provedou změny v rozvrhu, může tato změna uniknout bez povšimnutí a způsobit například neúčast na cvičení.

Kalendář v mobilu - zadávání všech hodin do kalendáře je poměrně časově náročné, ale představuje dobrou alternativu ke specializované mobilní aplikaci.

Kopírování do tabulkového editoru a následná editace.

Ruční přepisování na papír, nebo do poznámkového bloku.

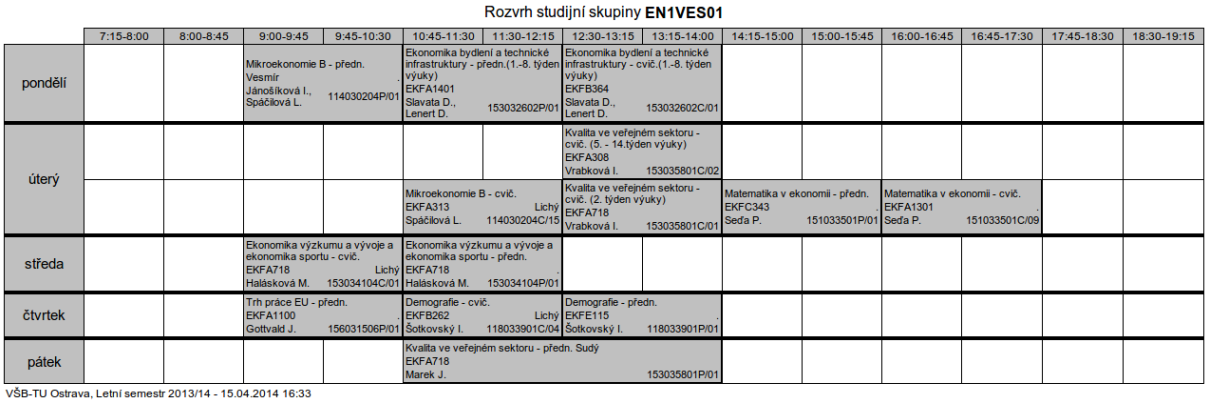
Někteří studenti spoléhají na svou **vlastní paměť** a na rozvrh se podívají doma před odchodem do školy, což může vést k zapomenutí, záměně časů a podobně.

Zobrazení na mobilním zařízení se věnujeme v další kapitole.

3.1.1 Dostupnost webové verze rozvrhů

Na webových stránkách školy jsou pro kohokoli dostupné všechny rozvrhy týkající se bakalářského a navazujícího studia pro formy prezenční, kombinovanou a celoživotní formu studia. Jejich zobrazení a vyhledávání v nich však není příliš vhodné pro mobilní přístroje, protože všechny ovládací prvky jsou příliš malé a špatně se ovládají. Ke komplexnosti navíc přispívá možnost zobrazení rozvrhů podle zadané školní skupiny, vyučujícího, místnosti, nebo názvu předmětu.

Obrázek 3.1 Webová verze rozvrhů



Rozvrh studijní skupiny **EN1VES01**

	7:15-8:00	8:00-8:45	9:00-9:45	9:45-10:30	10:45-11:30	11:30-12:15	12:30-13:15	13:15-14:00	14:15-15:00	15:00-15:45	16:00-16:45	16:45-17:30	17:45-18:30	18:30-19:15
pondělí			Mikroekonomie B - předn. Vesmír Jánošíková I., Spáčilová L. 114030204P/01		Ekonomika bydlení a technické infrastruktury - předn.(1.-8. týden výuky) EKFA1401 Slavata D., Lenert D. 153032602P/01		Ekonomika bydlení a technické infrastruktury - cvič.(1.-8. týden výuky) EKFB364 Slavata D., Lenert D. 153032602C/01							
úterý							Kvalita ve veřejném sektoru - cvič. (5. - 14.týden výuky) EKFA308 Vrabková I. 153035801C/02							
					Mikroekonomie B - cvič. EKFA313 Lichý Spáčilová L. 114030204C/15		Kvalita ve veřejném sektoru - cvič. (2. týden výuky) EKFA718 Vrabková I. 153035801C/01		Matematika v ekonomii - předn. EKFC343 Seďa P. 151033501P/01		Matematika v ekonomii - cvič. EKFA1301 Seďa P. 151033501C/09			
středa			Ekonomika výzkumu a vývoje a ekonomika sportu - cvič. EKFA718 Lichý Halásková M. 153034104C/01		Ekonomika výzkumu a vývoje a ekonomika sportu - předn. EKFA718 Halásková M. 153034104P/01									
čtvrtek			Trh práce EU - předn. EKFA1100 Gottvald J. 156031506P/01		Demografie - cvič. EKFB262 Lichý Šotkovský I. 118033901C/04		Demografie - předn. EKFE115 Šotkovský I. 118033901P/01							
pátek					Kvalita ve veřejném sektoru - předn. Sudý EKFA718 Marek J. 153035801P/01									

VŠB-TU Ostrava, Letní semestr 2013/14 - 15.04.2014 16:33

Zdroj: Rozvrh VŠB-TUO, 2014

3.1.2 Výjimky pro studijní skupiny

Aby vše nebylo příliš jednoduché, existují i zde výjimky v zobrazování rozvrhu pro studenty, kteří si svůj studijní plán tvoří sami, nicméně i pro ně by mohla mít aplikace význam, protože si zde mohou dohledat kdy a ve kterých učebnách se konkrétní předměty vyučují, případně jako pomocník při vyhledávání volné učebny.

3.2 Analýza požadavků

Důkladná analýza je důležitá pro úspěšný návrh a implementaci projektu. Opomenutí některých požadavků by mohlo mít za následek nižší kvalitu výsledného produktu. Požadavky musí být jasně a detailně definovány pro účely návrhu systému. (SCHWALBE, 2011)

Sběr požadavků – komunikace s budoucími uživateli za účelem získání přehledu o jejich požadavcích na systém.

Analýza požadavků – vyhodnotíme zjištěné informace a vyvodíme z nich závěry.

Dokumentace požadavků – tvorba textových dokumentů a grafů popisujících navrhovaný systém.

Pomocí rozhovorů se studenty byly získány následující požadavky, které by koncový uživatel mohl mít.

- Uživatelé chtějí mít rychlý přehled o vyučovaných hodinách ve své skupině.

- Někomu nemusí vyhovovat čas a den výuky a tak si bude chtít najít alternativní vyučovací hodinu v jiný čas, nebo den. Bylo by vhodné vypsat hodiny, kdy se vyučuje daný předmět, případně vypsat rozvrh místnosti.
- Své rozvrhy by chtěli vyhledávat studenti i vyučující.
- Studenti si chtějí sestavovat vlastní rozvrh, případně doplňovat volitelné hodiny.
- Rozvrhy se průběžně mění, dochází k přesunům vyučovacích hodin.

Analýzou těchto požadavků jsme došli k následujícím případům užití.

3.2.1 Podklady pro Android aplikaci

Školní rozvrhy můžeme uživateli nabídnout v Android aplikaci za předpokladu, že k nim budou dostupné podklady. V první fázi byla k dispozici jen webová verze rozvrhu na školním serveru. Analýzou těchto stránek jsme zjistili, že mají HTML strukturu takovou, ze které bude možné tato data získat. K tomuto účelu byla napsána jednoduchá aplikace pro parsování dat. Její funkce vypadala tak, že v cyklu načítala všechny webové stránky rozvrhů pro skupiny denního studia a z nich získávala podklady pro zobrazení rozvrhů. Takto získávat data bylo poměrně náročné na výkon a ne příliš efektivní vzhledem k tomu, že by tento skript měl běžet alespoň jednou denně.

Nejlepší řešení bylo zjistit, jestli by nebylo možné získat podklady přímo od školy v nějakém lépe zpracovatelném formátu. Požadovaná data, po konzultaci, přislíbil univerzitní rozvrhář pan Ing. Juráň, který byl velmi ochotný a data by mohl zařídit, pokud bude podána žádost. Podle domluvy byla tedy podána žádost oficiální cestou, které bylo následně vyhověno a potřebná data poskytnuta.

3.3 Případy užití Android aplikace

Use case diagram nám zobrazuje chování popisovaného systému tak, jak jej vnímá uživatel. Případ užití můžeme popsat jako sadu několika akcí, které vedou k dosažení požadovaného cíle, nepopisujeme zde jakým způsobem toho dosáhnout (FOWLER, 2004). Vzájemný vztah mezi uživatelem a vytvářenou Android aplikací si ukážeme na jednoduchém případu užití. Ten se skládá z následujících částí.

Účastník – reprezentuje kohokoli mimo popisovaný systém. Tato osoba se systémem komunikuje, přijímá, nebo vkládá informace.

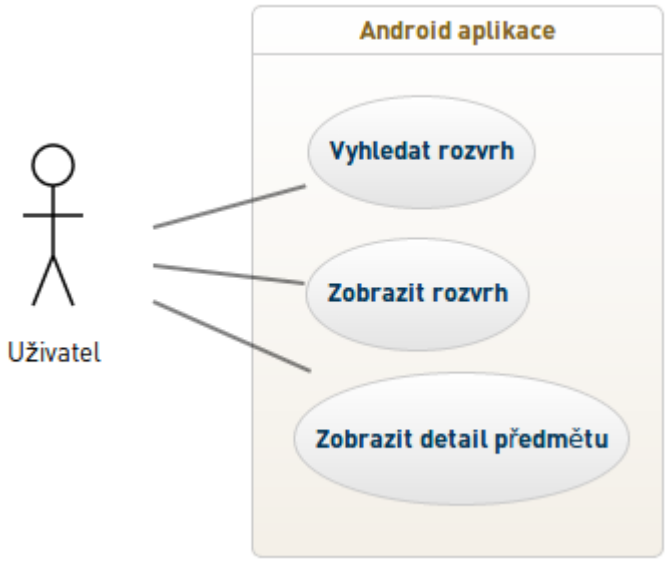
Typová činnost (use case) – charakterizuje konkrétní použití systému účastníkem.

Ohraničení – vymezuje hranice popisovaného systému.

Relace – vymezuje komunikaci mezi vnějším prvkem (účastníkem) a vnitřním případem užití.

Z analýzy uživatelských požadavků vytvoříme jednoduchý diagram. Obrázek Obrázek 3.2 Android aplikace rozvrhu - případ užití nám říká, že máme jen jeden typ uživatele, který má přístup ke třem činnostem v systému.

Obrázek 3.2 Android aplikace rozvrhu - případ užití



Zdroj: Vlastní zpracování

Vyhledání rozvrhu je potřeba provést vždy, když si chceme zobrazit jiný rozvrh, než máme aktuálně načten k zobrazení.

Zobrazení rozvrhu využijeme, při spuštění aplikace, kdy se nám zobrazí poslední navštívený rozvrh, nebo po volbě vyhledaného rozvrhu.

Zobrazení detailu předmětu umožňuje zjistit více informací o konkrétní vyučovací hodině.

Jak tyto případy užití navzájem spolupracují si popíšeme v následující kapitole.

3.4 Navigace v aplikaci

Aplikace se skládá ze dvou hlavních obrazovek, a to výpisu rozvrhu se záložkami a detailu vyučovaného předmětu. Základní navigace tak neobsahuje velké množství funkcí a uživatel se v ní nemůže jednoduše ztratit.

3.4.1 Záhlaví aplikace

V levém horním rohu aplikace se bude nacházet hlavní ikona, bude zobrazena za každé situace v každém okně. V pravém horním rohu bude tlačítko určené k vyhledání jiného rozvrhu než je aktuálně zobrazený.

3.4.2 Volba rozvrhu

Kliknutím na ikonu vyhledávacího tlačítka v pravém horním rohu, vybídne aplikace k zadání názvu hledaného rozvrhu (skupina, vyučující, místnost). Bude použita komponenta Zadáváním textu do vyhledávacího políčka se okamžitě vypíší nalezené rozvrhy k zobrazení. Po kliknutí na požadovaný název rozvrhu se načte obsah v podobě vyučovaných předmětů v jednotlivých dnech.

Pro zadávání vyhledávání bude využita komponenta SearchView, která standardně podporována od verze API 11, ale díky podpůrné knihovně Support Library ji budeme moct využít i pro API verze 8 a vyšší (Android developers, 2014). To naši aplikaci omezí jen nepatrně, protože uživatelů s těmito staršími zařízeními je velmi málo (jak nám ukázal obrázek 2.2).

3.4.3 Přepínání mezi dny v týdnu

Mezi dny v týdnu bude možné se přesouvat pomocí gesta swipe, nebo kliknutím na název dne v záhlaví. Umožní nám to standardní komponenta ViewPager.

3.4.4 Detail předmětu

Kliknutím na konkrétní předmět ve výpisu rozvrhu, se zobrazí další doplňující informace k této položce, jako jsou konkrétní dny vyučování a seznam vyučujících. Zpátky na rozvrh hodin se dá dostat pomocí systémového navigačního tlačítka *zpět*, nebo kliknutím na ikonu

aplikace, kde bude v této situaci zobrazena šipka vlevo, což značí návrat k předchozí obrazovce.

3.5 Podpůrná webová aplikace

Zpracovaná data budeme poskytovat z webové služby, která bude neustále dostupná ze sítě Internet. Bude zpracována pomocí jazyka Java.

Ve své podstatě se jedná o jednoduchou aplikaci, která má dva hlavní úkoly.

- **Získání rozvrhových dat** zahrnuje stažení dat poskytnutých univerzitními rozvrháři a jejich uložení do relační databáze.
- **Poskytování dat pro aktualizace** zajišťuje webová služba dostupná odkudkoli z Internetu ve formátu JSON.

Data se budou automaticky aktualizovat v noci jednou denně. Častěji by to nemělo význam, protože poskytovaná data se rovněž generují jednou denně.

3.6 Návrh databáze

Pomocí analýzy požadavků a vstupních dat navrhneme databázové schéma pro uložení importovaných dat. Pro naši potřebu nebudeme ukládat veškerá poskytnutá data, ale jen ta, která v aplikaci využijeme. Ukládání veškerých dat by zapříčinilo větší objem databáze a v důsledku také pomalejší aktualizace, vyšší objem přenášených dat.

Proces návrhu se skládá z několika následujících kroků, které nám pomůžou vytvořit správnou strukturu databáze. (SCHNEIDER, 2006)

Určíme si účel databáze – data zde budou uložena pro potřeby synchronizace s Android aplikací. Jedná se o centrální databázi, kterou budou využívat všichni uživatelé aplikace.

Zjistíme jaké informace budou uloženy – analyzujeme vstupní data z CSV souboru a určíme datové typy jednotlivých sloupců. Vynecháme nepotřebné údaje.

Rozdělíme data do tabulek – pro naše potřeby budeme potřebovat tabulku předmětů, vyučujících, studijních skupin, místností a vyučovacích hodin.

Definujeme jednotlivé sloupce tabulek – z analýzy dat v CSV souboru určíme názvy sloupců a datové typy.

Určíme primární klíče – každá tabulka bude mít svůj primární klíč, v tomto případě vždy půjde o automaticky inkrementované číslo.

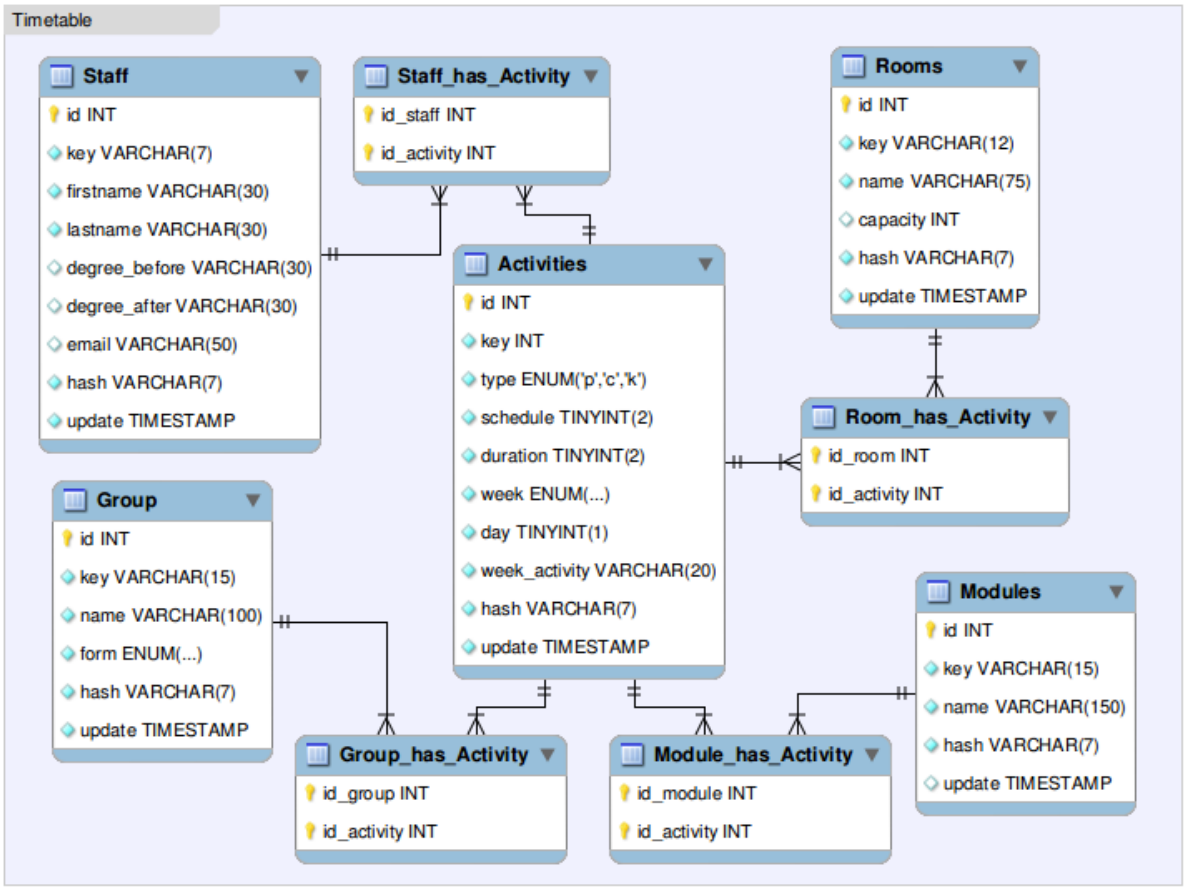
Určíme jednotlivé vazby mezi datovými tabulkami – v tomto případě půjde vždy o vazby M:N mezi souvisejícími tabulkami.

Kontrola návrhu – provedeme ověření správného návrhu a vložíme zkušební data.

Normalizujeme databázi – aplikujeme normalizační pravidla, čímž získáme správně strukturovaný konzistentní návrh bez redundantních dat. Zbytečně velké množství dat by mohlo způsobit pomalejší načítání na straně Android aplikace.

Pro snadnější orientaci v návrhu používáme ER model, na němž snáze vidíme jednotlivé vazby mezi relacemi.

Obrázek 3.3 Relační model databáze MySQL



3.7 Databáze v telefonu

Pro uložení dat je využita standardní databáze SQLite, která je dostupná pro všechny verze systému Android. Data poskytnutá podpůrnou webovou aplikací se synchronizují s touto databází a obsahují tak stejnou strukturu dat.

Hlavním důvodem uložení všech dat v telefonu je, aby aplikace byla plně funkční při nedostupnosti datového připojení a studenti se tak mohli kdykoli podívat na svůj aktuální rozvrh.

Kompaktní databáze SQLite nemá shodné datové typy jako MySQL a tak budeme muset v některých případech využít alternativy. To nebude mít na celkovou funkčnost žádný vliv.

3.8 Popis webových služeb

V příkladech si popíšeme navrhovanou implementaci konkrétních služeb. Popis obsahuje volání konkrétním typem metody a název volané služby. Následně, pokud se služba vykoná bez chyb, je vrácena odpověď se statusem číslo 200 (v pořádku vykonáno).

3.8.1 Služba – uvítací zpráva

Po spuštění Android aplikace se zobrazí uvítací zpráva, která bude načítat text z této služby. Své využití bude mít při důležitých upozorněních a novinkách. Poskytovaný obsah může být i ve formátu HTML, není však podmínkou.

```
Požadovaná implementace:

Požadavek:
GET /message

Odpověď:
Status: HTTP/1.1 200 OK
Tělo odpovědi:
Textové sdělení.
```

3.8.2 Služba – informace o semestru

Poskytuje informaci o začátku a konci aktuálního semestru, případně semestru následujícího. Tato informace bude využita pro potřeby výpočtu konkrétních vyučovacích dní, které nejsou v konečné podobě k dispozici v poskytovaných datech. Vždy by měla být k dispozici informace minimálně o jednom semestru a to buď aktuálně probíhajícím, nebo nově nastávajícím. Již ukončené semestry zde nejsou nadále potřeba.

Android aplikace by si měla uchovávat tuto hodnotu ve své paměti a ověřovat ji jen zřídka (bude stačit jednou týdně). Pokud nastane chyba v tomto dotazu, klientská aplikace by měla využít stávající informaci, kterou získala v předchozím úspěšném volání této služby.

```
Požadovaná implementace:

Požadavek:
GET /semester

Odpověď:
Status: HTTP/1.1 200 OK
Tělo odpovědi:
{
    "semesters": [
        {
            "type": "winter",
            "start": "2013-09-15",
            "end": "2013-12-09"
        },
        {
            "type": "summer",
            "start": "2014-02-10",
            "end": "2014-07-15"
        }
    ]
}
```

3.8.3 Služba – rozvrhy hodin

Služba přijímá od Android aplikace seznam hash kódů, jejichž pomocí se při porovnání s databází na webové službě zjistí, která data byla změněna, přidána, nebo odebrána. Služba identifikuje položky, které je třeba smazat a jejich seznam vrátí ve výsledku spolu s nově přidanými záznamy. Android aplikace by měla podle těchto získaných dat aktualizovat svou lokální databázi. Tato služba bude nejnáročnější, co se přenosu většího objemu dat týče. Požadavek by se měl vykonávat jednou denně, aby nepřetížil tuto webovou službu při větším počtu instalací klientské aplikace.

Pokud by se vyskytla chyba, nepošle se žádný obsah, Android aplikace proces aktualizace zruší a bude opakovat požadavek později. Při opakovaných pokusech by měla být dostatečná prodleva tak, aby se najednou nezačalo generovat velké množství dotazů a nezpůsobilo to DOS útok na vlastního poskytovatele dat.

```
Požadovaná implementace:

Požadavek:
POST /schedules
Tělo požadavku:
{
```

```
    "modules": [
        {
            "hash": "c5e93ab"
        },
        {
            "hash": "5a780b3"
        },
        {
            "hash": "de16e5a"
        }
    ]
}
```

Odpověď:

```
Status: HTTP/1.1 200 OK
Tělo odpovědi:
{
    "version": "v1",
    "timestamp": "2014-03-15",
    "modules": {
        "add": [
            {
                "name": "Statistika A",
                "code": "151030301"
            },
            {
                "name": "Informační systémy",
                "code": "545010208"
            }
        ],
        "delete": [
            "de16e5a",
            "5a780b3"
        ]
    }
}
```

3.8.4 Služba – status

Zjišťuje, zda jsou hlavní komponenty aplikace v pořádku a poskytují správné odpovědi. Dále vypisuje aktuální verzi Android aplikace. Bude použito pro manuální zjištění, zda uživatel aktualizuje Android aplikaci v Google Play a z toho může Android aplikace vyvodit důsledky.

Výskyt chyby nebude mít žádný vliv na chod Android aplikace, ale měl by být okamžitě řešen ze strany správce webové služby. Nesprávná funkčnost této služby bude pravděpodobně znamenat problém i ostatních poskytovaných služeb. Při problémech by se měl automaticky upozornit správce, například zasláním zprávy na e-mail, nebo textové zprávy na telefon.

```
Požadovaná implementace:

Požadavek:
GET /status

Odpověď:
```

```
Status: HTTP/1.1 200 OK
Tělo odpovědi:
{
    "version": "v1",
    "code": "1.0.0",
    "updated": "2014-01-23 03:15:00",
    "android": {
        "apk": "0.2.1",
        "released": "2014-01-25"
    },
    "health": {
        "space": "OK",
        "database": "OK"
    }
}
```

3.8.5 Služba – statistiky

Služba vypíše počty záznamů v klíčových tabulkách. Slouží pro kontrolu, zda webová aplikace obsahuje data. Je zde jen pro potřeby správce webového serveru. Není potřebná k chodu Android aplikace.

```
Požadovaná implementace:

Požadavek:
POST /stats

Odpověď:
Status: HTTP/1.1 200 OK
Tělo odpovědi:
{
    "activities": 12123,
    "staff": 123,
    "rooms": 500,
    "groups": 900
}
```

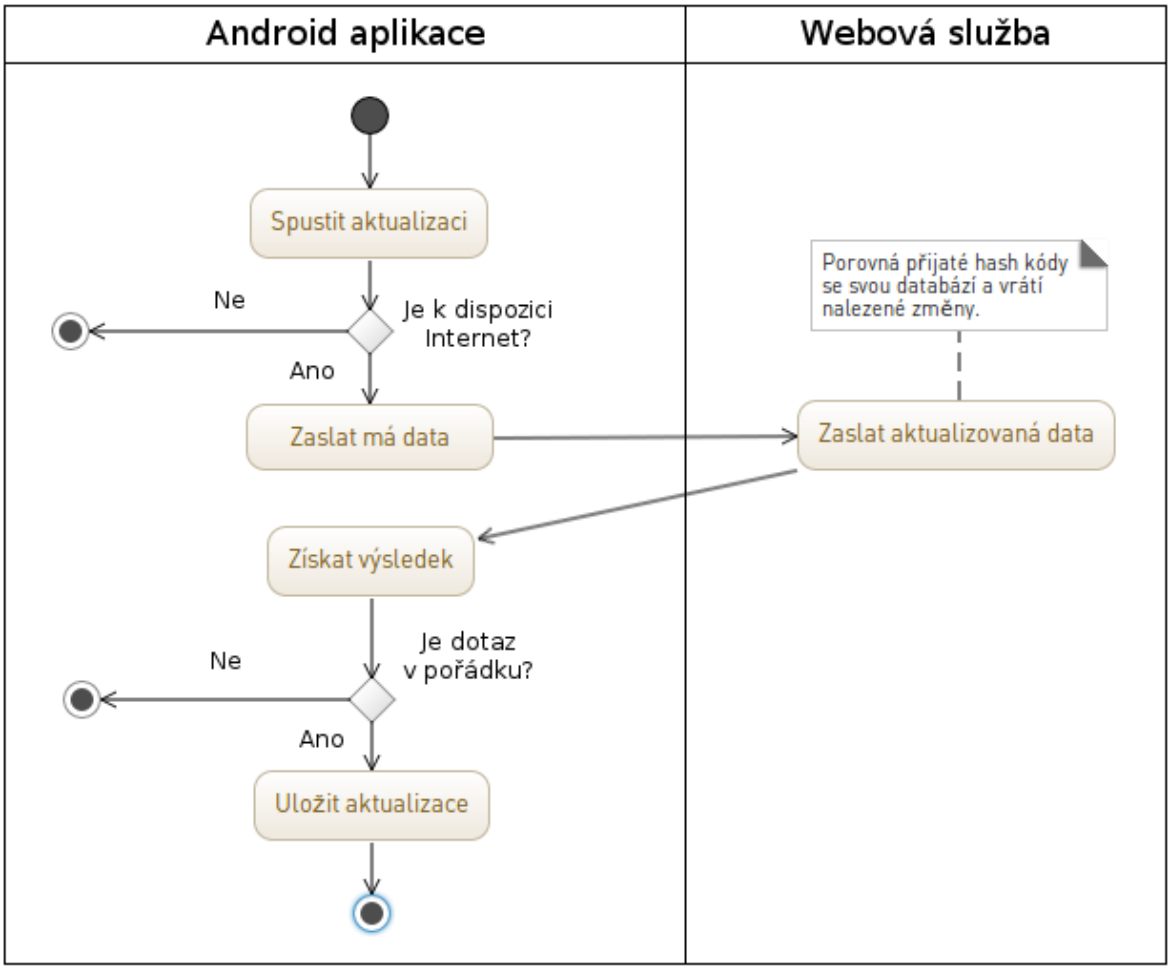
3.9 Proces aktualizace dat v Android aplikaci

Diagram aktivit zobrazuje dynamickou stránku popisovaného systému. Každý proces je zde realizován sekvencí kroků, vedoucí k určitému cíli. Představuje jakým způsobem se naše aplikace bude chovat vůdči podpůrné službě poskytující data.

Spuštění procesu aktualizace provede služba v pozadí, která je naplánována k provádění tohoto úkolu jednou denně. Nejprve si ověří, zda je k dispozici přístup do sítě Internet. Pokud ne, aktualizace se ukončí. Jestliže je přístupný, vytvoří požadavek na aktualizaci dat, odešle jej webové službě a ta mu zpět vrátí nové a modifikované položky ze své centrální databáze. Android aplikace přijme výsledek svého dotazu a pokud je v pořádku, uloží aktuální data do své lokální databáze, v opačném případě ukončí aktualizaci.

Druhou možností spuštění tohoto procesu je, že při aktivaci Wi-Fi připojení si Android aplikace zjistí kdy provedla poslední úspěšnou aktualizaci a pokud je to více jak jeden den, zahájí aktualizační proces.

Obrázek 3.4 Aktivity diagram - aktualizace databáze v Android zařízení



3.10 Grafika aplikace

Vizuální prvky se budou zobrazovat na zařízeních, které mají různé rozlišení a úhlopříčky displeje. Potřebujeme tedy vyřešit, aby zobrazované obrázky nebyly na displejích s velkým počtem obrazových bodů příliš malé.

Grafika pro mobilní zařízení musí být flexibilní, aby se mohla přizpůsobovat různým rozlišením displeje. Pro tyto potřeby je vhodné využít především textové definice v XML. Takto zadané grafické prvky se dokážou přizpůsobovat zařízením podle potřeby a jejich velikost je menší, než kdyby byly přiloženy jako bitmapové soubory. Problém rastrové

grafiky je především v tom, že musíme vytvářet samostatné obrázky pro různá rozlišení, což ve výsledku zvětšuje velikost aplikace a je komplikovanější na údržbu.

Grafické zpracování bude zvoleno tak, aby barevně ladilo se systém Edison. Hlavní dominantní barva je zde zelená a pozadí bílé. Ikona aplikace se také inspiruje vzhledem loga tohoto systému.

Obrázek 3.5 Příklad vytváření bitmapové grafiky

Zdroj: Android developers, 2014

4 Implementace mobilní aplikace a datového serveru

Jako první byl vytvořen základní prototyp aplikace, která zvládala všechny základní činnosti. To bylo důležité pro ověření, zda je zamýšlené řešení provozuschopné tak, jakým způsobem bylo navrženo. Při vývoji se postupovalo po jednotlivých menších krocích, kde na konci každého byla funkční aplikace.

4.1 Nástroje použité pro vývoj aplikace

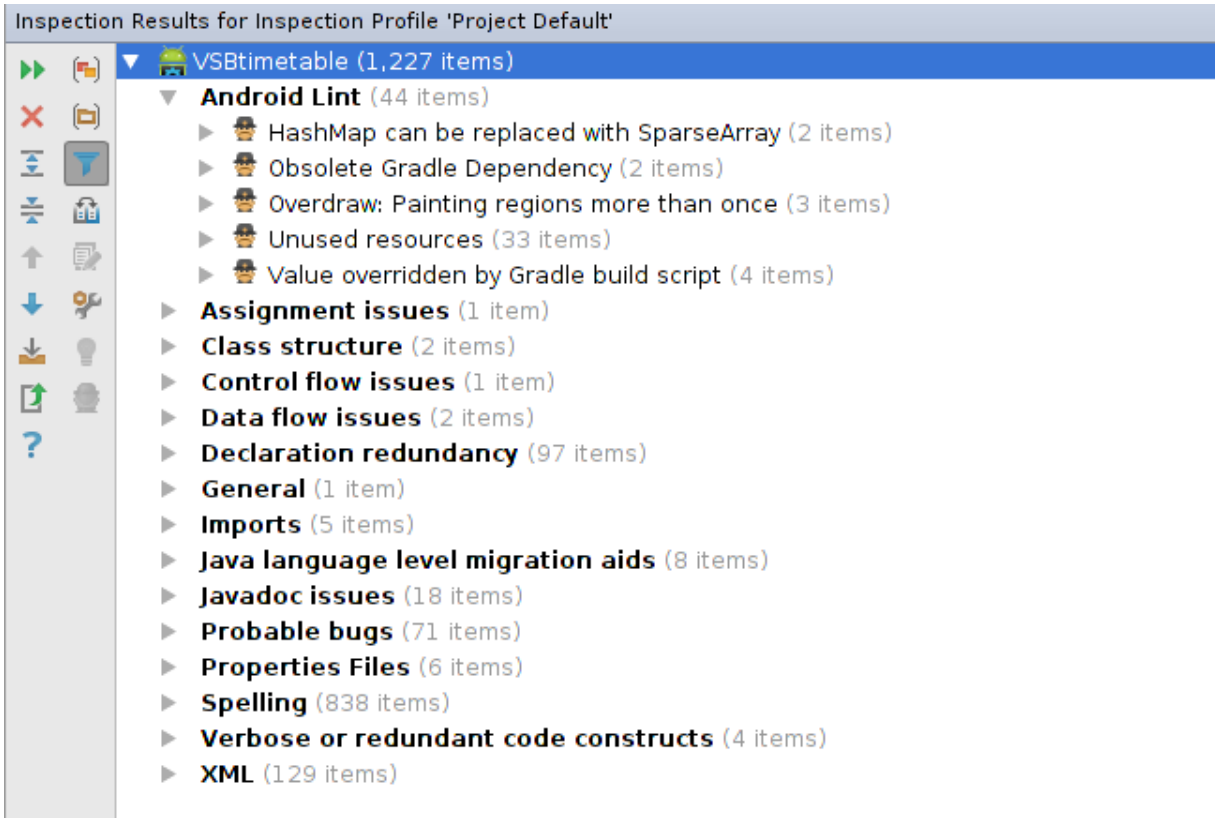
4.1.1 Vývojové prostředí AndroidStudio

Jako vývojového prostředí bylo využito IDE AndroidStudio (založeno na IntelliJ IDEA), které je stále v předprodukční verzi, ale již dnes v mnoha směrech předčí aktuálně oficiální IDE Eclipse s pluginem ADT (Android Developer Tools). Hlavní důvody jsou především v lepší uživatelské přívětivosti, důkladné analýze kódu a využití nástroje Gradle pro sestavení spustitelné verze aplikace ze zdrojových kódů.

Editor průběžně provádí inspekci kódu a případně upozorňuje na nesprávný zápis. Velmi užitečná je kontrola inicializace proměnných a varování na neošetřené výjimky v kódu. Na vyžádání můžeme spustit hloubkovou analýzu, kde zjistíme místa, která bychom měli ošetřit, protože by mohla být potenciálním zdrojem problémů.

Na obrázku vidíme, jak rozsálé zkoumání tohle vývojové prostředí provádí. Dává nám velké množství užitečných rad, jak náš kód optimalizovat. Ne všechno musí být nutně chyba, ale některé z těchto položek mohou snižovat kvalitu naší aplikace.

Obrázek 4.1 Podrobná inspekce kódu



Zdroj: Vlastní zpracování

4.1.2 Vývojové prostředí NetBeans

Podpůrná webová aplikace je vytvářena v prostředí NetBeans, které obsahuje veškeré potřebné nástroje pro efektivní tvorbu webových služeb založených na jazyce Java.

4.2 Podpůrná webová aplikace

Webová aplikace je nasazena na veřejném serveru a je dostupná 24 hodin denně. Je napsána v jazyce Java a využívá REST služeb k poskytování dat pro Android aplikace.

Na obrázku vidíme, jak jednoduše se dají v Javě implementovat webové služby.

Obrázek 4.2 Příklad implementace webové služby

```
@GET
@Path("/semester")
@Produces(MediaType.APPLICATION_JSON)
public Response semester() throws JSONException, IOException {

    // Get semester dates
    JSONObject results = timetableBean.semester();

    return Response
            .status(200)
            .entity(results.toString(4)).build();
}
```

Zdroj: Vlastní zpracování

4.2.1 Získání rozvrhových dat

Stažení dat poskytnutých univerzitními rozvrháři a jejich uložení do relační databáze. Každý den se v nočních hodinách spouští skript pro stažení poskytnutých dat univerzitními rozvrháři.

Komprimovaná data mají velikost přibližně 200 kB a jsou k dispozici na následující adrese.

```
http://edison.sso.vsb.cz/export/scheduleExport.zip
```

Rozbalením tohoto archivu dostaneme 10 souborů v textovém formátu CSV, jejichž velikost je přibližně 1 MB. Připravená data se budou pravidelně načítat do relační databáze.

Obrázek 4.3 Příklad vstupních dat - CSV formát

```
"PAN0017";"Panec";"Alan";"Mgr.";"";"alan.panec@vsb.cz"
"PAN083";"Panovec";"Vladan";"Ing.";"";"vladan.panovec@vsb.cz"
"PAN37";"Pánek";"Petr";"doc. Ing.";"CSc.";"petr.panek@vsb.cz"
"PAP0032";"Pápeš";"Martin";"Ing.";"";"martin.papes@vsb.cz"
```

Zdroj: Data poskytnutá univerzitními rozvrháři

Aktualizace probíhá tak, že každý záznam z tabulky má svůj vygenerovaný hash kód a podle něj se pozná, zdali se daná položka změnila, či nikoli.

Obrázek 4.4 Zpracovaná data v relační databázi

edit	id	key	lastname	firstname	title_before	title_after	email	hash	update
edit	974	PAN083	Panovec	Vladan	Ing.		vladan.panovec@vsb.cz	c772a5b	2013-10-05 16:36:45
edit	975	PAN37	Pánek	Petr	doc. Ing.	CSc.	petr.panek@vsb.cz	e481642	2013-10-05 16:36:45

Zdroj: Vlastní zpracování

4.2.2 Poskytování dat pro aktualizace

Generuje data ve formátu JSON podle poskytnutých parametrů v dotazu na tuto službu. Je dostupná kdykoli a výsledný vygenerovaný obsah se liší podle aktuálnosti klientské Android aplikace.

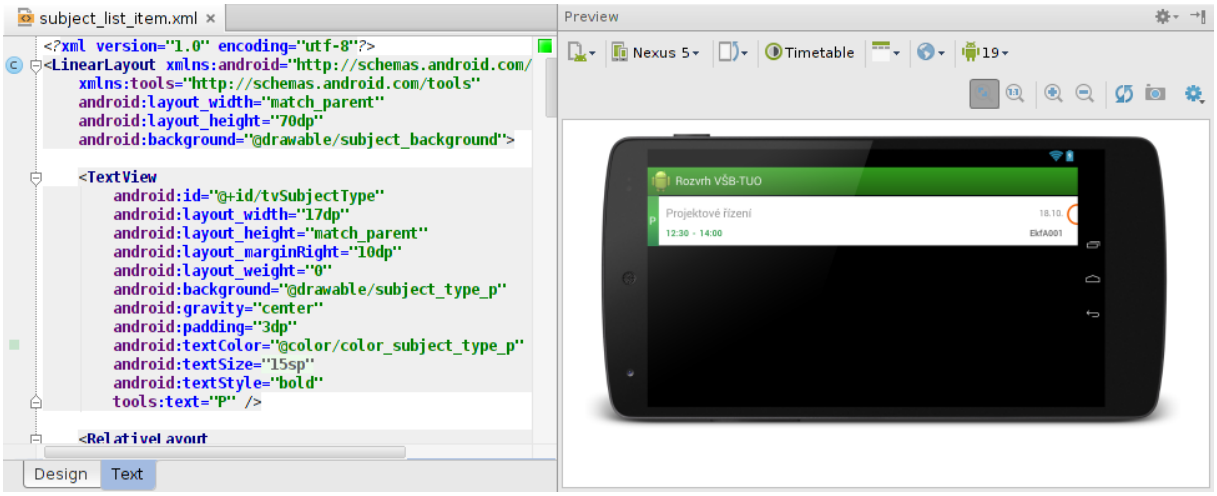
Do budoucna by bylo vhodné využít cache paměť na uložení opakujících se dotazů a tím ke snížení zátěže této služby.

4.3 Implementace Android aplikace

4.3.1 Vytváření grafických prvků

Zpracování grafických prvků mi velmi usnadnil editor AndroidStudio. Díky živému náhledu vytvářeného grafického layoutu můžeme vidět výsledek okamžitě v okně editoru. Není tak potřeba aplikaci neustále spouštět, abychom viděli, jak se naše změny v kódu projeví. Na následujícím obrázku je vlevo vidět editovaný kód v jazyku XML a jeho výsledek, pro lepší představu i s rámečkem telefonu okolo. Velkou výhodou je, že si můžeme zvolit velikost úhlopříčky zařízení v náhledu a jeho rozlišení. Efektivně tak lze odladit layout pro velké množství různých zařízení bez toho, abychom je museli vlastnit, nebo si pro každý případ zvláště nastavovat a spouštět emulátor.

Obrázek 4.5 AndroidStudio - editace rozložení položky seznamu



Zdroj: Vlastní zpracování

4.3.2 Informace o Android aplikaci – AndroidManifest.xml

Jedná se o soubor popisující základní stavební kameny naší aplikace. Pro systém android definuje klíčové prvky pro celý systém.

Verze – celočíselná hodnota, slouží jako primární identifikace nové verze a musí být vždy vyšší než předchozí verze. Dále je zde kódové označení verze, které může mít volitelný formát.

Verze API – určuje úrovně programové platformy, pro které je tato aplikace určena. Definuji zde minimálně Android ve verzi 2.2, což je API level 8.

Oprávnění k využití systémových prostředků – žádáme o oprávnění k využití internetového připojení pro potřeby aktualizací dat.

Spouštěcí místo aplikace – říká systému, kterou aktivitu aplikace má zavolat, pokud se vyskytne požadavek na spuštění nové instance aplikace.

Ikona aplikace – poskytuje spouštěcí ikonu pro seznam nainstalovaných aplikací v systému.

Služby – obsahuje seznam služeb, které aplikace může spouštět v pozadí.

Přístupové body – samotná aplikace může poskytovat data jiným aplikacím, zde definujeme potřebné poskytovatele těchto dat.

4.3.3 Jednotkové testování aplikace

Během vytváření aplikace vzniká velké množství programového kódu, abychom udrželi klíčové komponenty bez chyb, využili jsme automatizované testování Android aplikace.

Obrázek 4.6 Testování klíčových metod

```
@SmallTest
public void testBadParameterFormatDay() {
    try {

        WeekDate weekDate = new WeekDate("001000000", 7);
        Assert.fail();
    } catch (InvalidParameterException e) {
        // success
    }

    try {

        WeekDate weekDate = new WeekDate("001000000", 0);
        Assert.fail();
    } catch (InvalidParameterException e) {
        // success
    }
}
```

Zdroj: Vlastní zpracování

4.3.4 Knihovna Volley

Volley je knihovna, která umožňuje propojení Android aplikace s webovými službami jednodušší a hlavně rychlejší cestou.

Mezi její hlavní výhody patří:

- Automaticky naplánuje všechny síťové požadavky z aplikace.
- Poskytuje transparentní ukládání do dočasné paměti.
- Poskytuje možnost zrušení volání API, například pokud nám webová služba neodpovídá.
- Nabízí nástroje na ladění a trasování dotazů.

Na obrázku lze vidět jednoduché volání vzdálené služby. Tu nějtěžší práci udělá knihovna za nás a my se můžeme soustředit na důležitější části kódu.

Obrázek 4.7 Volání vzdálené služby

```
protected void downloadData() {

    RequestQueue queue = Volley.newRequestQueue(context);

    JsonObjectRequest jsObjRequest = new JsonObjectRequest(Request.Method.POST,
            DOWNLOAD_SERVER_URI, createRequestData(), (response) -> {

            update(response);
    }, new Response.ErrorListener() {

        @Override
        public void onErrorResponse(VolleyError error) {
            Log.w(TAG, "Aktualizace selhala: " + error.getMessage());
        }
    });

    queue.add(jsObjRequest);
}
```

Zdroj: Vlastní zpracování

4.3.5 Vyhledávání rozvrhu

Po kliknutí na tlačítko pro vyhledávání, se bude zobrazovat nabídka položek, které vyhovují zadávanému řetězci. Následující pohled na data zajišťuje sjednocení názvů, podle kterých budeme vyhledávat. Čím více znaků zadáme do textového pole, tím méně dostaneme výsledků, budou tedy přesnější. Jak je vidět v SQL dotazu, budeme vyhledávat v následujících sloupcích klíčových tabulek.

Groups (Skupiny) – kód skupiny a název

Modules (Předměty) – kód předmětu a název

Staff (Zaměstnanci) – kód učitele a celé jméno včetně příjmení

Rooms (Místnosti) – kód místnosti a název

```
CREATE VIEW V_Search AS
SELECT "g" || `id` as type, `key` as text1, `name` as text2 FROM Groups
UNION
SELECT "m" || `id` as type, `name` as text1, `key` as text2 FROM Modules
UNION
SELECT "s" || `id` as type, `title_before`||' '||`firstname`||'
'||`lastname`||' '||`title_after` as text1, `key` as text2 FROM Staff
UNION
SELECT "r" || `id` as type, `key` as text1, `name` as text2 FROM Rooms;
```

4.3.6 Verzování aplikace

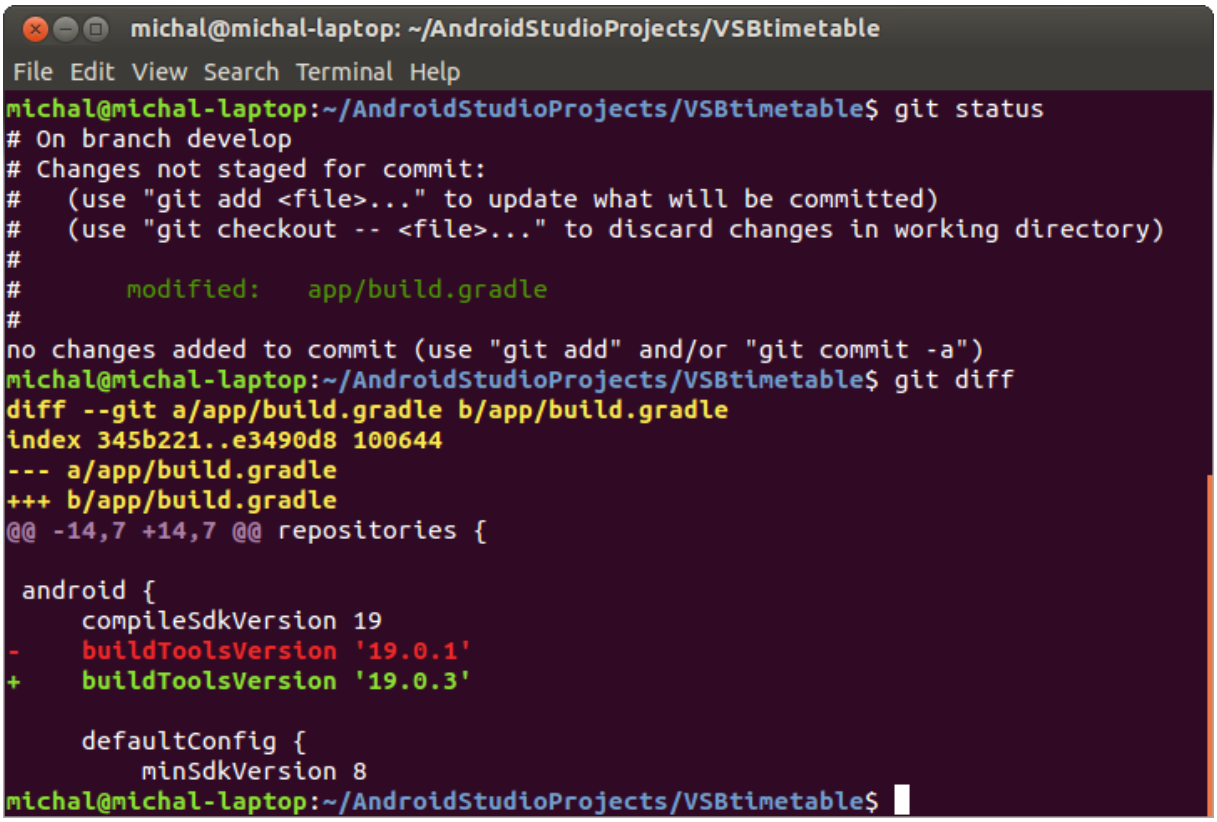
Změny v programovém kódu jsou průběžně zaznamenávány pomocí nástroje na správu verzí. K tomu zde využíváme distribuovaný verzovací systém GIT. Tento mocný nástroj je užitečný především při práci na větších projektech, ale zálohovat se vyplatí každý projekt, ať už je jeho rozsah jakýkoli. (GIT-SCM, 2014)

Všechny úpravy Android aplikace i webových služeb, se průběžně zaznamenávají a navíc se vše zálohuje na jiné místo na disku. Je to dobrý pomocník při revizích kódu. Díky kontrole před každým uložením kódu do archivu zde zanáší méně neoptimalizovaného kódu.

Velmi jednoduše se dá naučit, je rychlý a dobře se s ním pracuje. To se hodí především tehdy, kdy se vývoj ocitne ve slepé uličce a je třeba se vrátit ke starší verzi aplikačního kódu. Pro nové vlastnosti aplikace se využívají samostatné větve, takže můžeme pracovat na více nových funkcích najednou a stále mít pod kontrolou starý kód.

Typicky je využit z prostředí systémového příkazového řádku pro lepší kontrolu nad použitými operacemi. Je možné jej využít taky z prostředí AndroidStudio a NetBeans.

Obrázek 4.8 Správa verzí aplikace



Zdroj: Vlastní tvorba

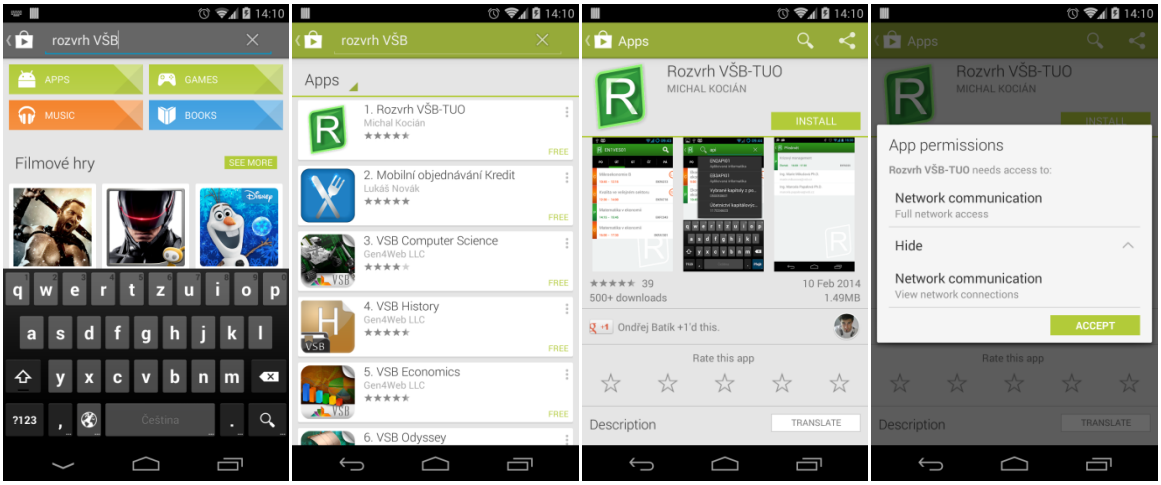
Obrázek ukazuje, jak vypadá typická práce s tímto nástrojem v příkazové řádce. Vše je pěkně přehledné a barevně odlišené.

4.4 Instalace aplikace

Dále popíšeme postup jak nainstalovat Android aplikaci z pohledu koncového uživatele. Způsobů jakými toho docílit je více, první uvedeme instalaci z telefonu a poté z webové stránky obchodu Google Play.

Na mobilním zařízení si spustíme aplikaci Google Play. Pro vyhledávání klikneme vpravo nahoře na symbol lupy, který nás vybídne k zadání textu. Vložíme například název „rozvrh VŠB“ a potvrdíme vyhledávání. Zobrazí se seznam relevantních aplikací a my zvolíme „Rozvrh VŠB-TUO“. Dostaneme se do detailu naší aplikace, kde se dozvíme podrobnější informace. Můžeme zde nalézt autora, náhledy vzhledu, hodnocení uživatelů, popis aplikace, nově přidané funkce, přibližný počet instalací, velikost aplikace, označení verze a datum poslední aktualizace programu. Kliknutím na tlačítko Instalovat se nám zobrazí informační dialog o požadovaných právech přístupu ke zdrojům. Po potvrzení se aplikace stáhne, nainstaluje do telefonu a bude připravena k použití.

Obrázek 4.9 Google Play - instalace aplikace



Zdroj: Vlastní zpracování

Další možností je přejít na webovou stránku obchodu Google Play (odkaz je uveden níže). Zde je možné kliknout na instalaci aplikace. Po přihlášení do Google účtu máte možnost zvolit, do kterého ze svých Android zařízení chcete aplikaci nainstalovat. Po potvrzení se

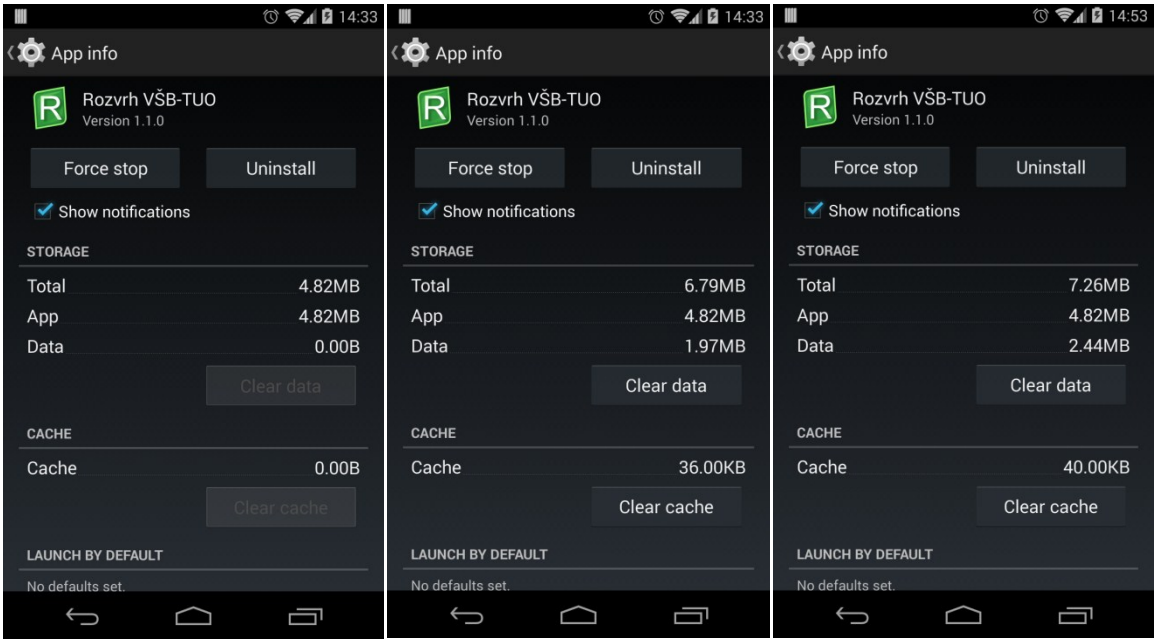
aplikace do zařízení okamžitě nainstaluje (pokud je aktuálně připojeno k Internetu). Můžete tak uskutečnit instalaci, aniž byste se Android telefonu dotkli.

```
https://play.google.com/store/apps/details?id=name.kocian.vsb.timetable
```

4.4.1 Velikost aplikace

Vygenerovaný instalační balíček aplikace APK má velikost přibližně 1.5 MB, z čehož 1 MB zabírají výchozí rozvrhová data. Instalací se aplikace zavede do systému a tím naroste na necelých 5 MB, stane se tak dostupná pro spuštění uživatelem. V této chvíli ještě nejsou žádná data v databázi, což ukazuje následující obrázek vlevo. Po prvním spuštění se inicializuje databáze, která do té doby byla připravena v podpůrných souborech balíčku, tím se rozbalí data o velikosti přibližně 2 MB, obrázek uprostřed. Ihned poté je může uživatel aplikaci využívat. Databáze se bude postupem času sama na pozadí aktualizovat a její velikost měnit, obrázek vpravo.

Obrázek 4.10 Informace o aplikaci



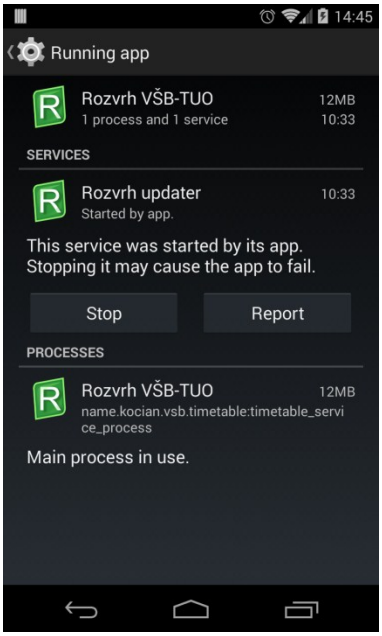
Zdroj: Vlastní zpracování

4.4.2 Využití operační paměti

Pro svůj běh vyžaduje 10 MB operační paměti. To je v porovnání například s aplikací Facebook, která vyžaduje 70 MB, docela dobrá hodnota.

Na pozadí běží jeden proces „Rozvrh updater“, který se stará o aktualizace na pozadí. Čeká na svou příležitost a nezabírá téměř žádné systémové prostředky. Jeho aktivace je naplánována jen jednou denně, více by nemělo být potřeba.

Obrázek 4.11 Využití paměti



Zdroj: Vlastní zpracování

4.4.3 Stažení aktuálních dat

Každý den v nočních hodinách je nastaveno provedení stažení aktualizovaného ZIP archivu, který serverová aplikace následně dekomprimuje, tím získá soubory CSV, ve kterých jsou obsažena veškerá potřebná data, jako názvy studijních skupin, seznam vyučujících, místnosti určené k výuce a vyučované předměty.

Datové soubory se následně okamžitě zpracují. Podpůrná aplikace si je všechny načte, ověří vazby mezi jednotlivými daty a poté vloží do databáze tak, že první smaže data, která již v nové aktualizaci nejsou, následně vloží nová data s tím, že se nezměněná data nepřepisují.

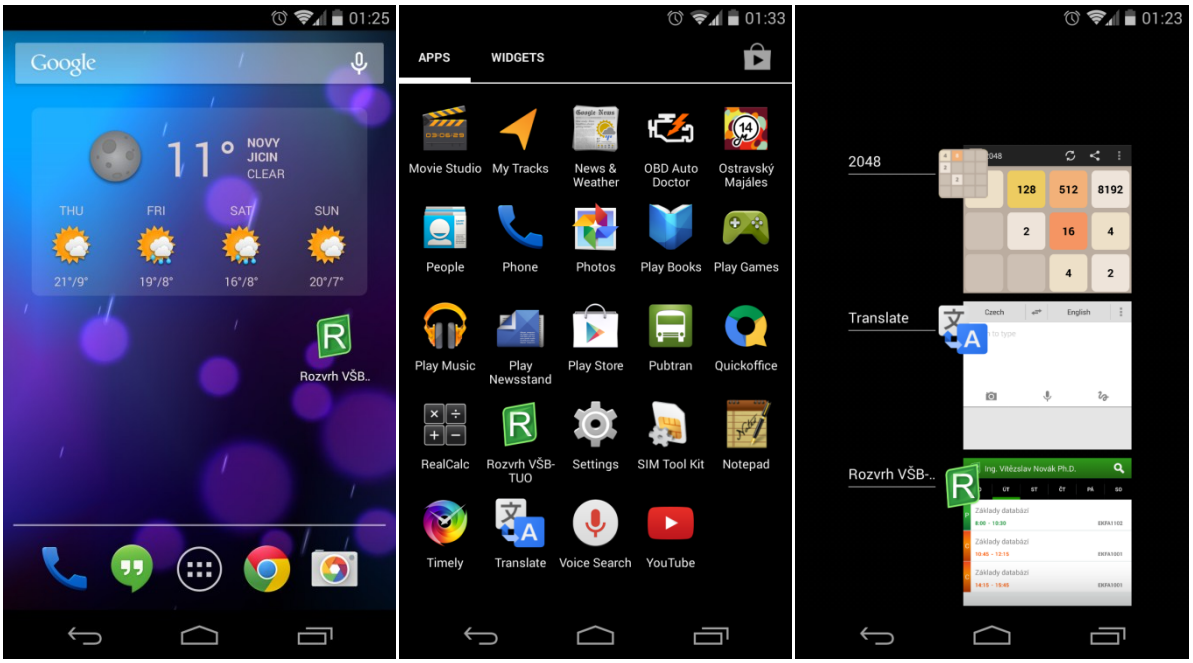
Aktualizace se spouští na pozadí a není tak potřeba, aby aplikace byla spuštěna.

4.4.4 Aplikace v prostředí systému

Podle ohlasů se aplikace po vizuální stránce poměrně povedla. Následující obrázky uvedou, jak se ji podařilo zasadit do systému ve srovnání s dalšími aplikacemi. Na různých zařízeních může být dojem mírně odlišný.

Výrobci telefonů většinou přizpůsobují uživatelské prostředí podle vlastního uvážení, je tedy nemožné s jistotou říci, že ikona aplikace bude vždy dobře viditelná a rozpoznatelná. Na druhou stranu její výrazná zelená barva by mohla vynikat téměř vždy, protože lze těžko předpokládat, že by tento odstín byl v systému využit častěji. Uživatelé mohou mít na hlavní obrazovce vlastní obrázkové tapety, kde je docela pravděpodobné, že by se zelená mohla ve větší míře vyskytovat, ale tomu se nedá nikdy úplně vyhnout.

Obrázek 4.12 Vzhled ikony v porovnání s ostatními aplikacemi



Zdroj: Vlastní zpracování

4.5 Nasazení podpůrné webové aplikace

Výsledná webová aplikace je nasazena na veřejném serveru, odkud k ní mají přístup všechna Android zařízení. Pokud by server nebyl dostupný, nebude to problém, Android aplikace se pokusí o synchronizaci dat znovu za několik hodin.

4.6 Publikace výsledné Android aplikace

Zpřístupnit aplikaci můžeme v jednom ze specializovaných obchodů pro Android aplikace. Je možné výslednou aplikaci odeslat přímo uživateli, ale tímto způsobem by bylo poměrně náročné dostat náš produkt k většímu počtu zájemců.

4.6.1 Vývojářský účet Google

Publikovat aplikaci v Google Play můžeme jen za předpokladu, že vlastníme vývojářský účet Google. Ten je přístupný po registraci a zaplacení jednorázového poplatku 25$. Zde pomocí webového rozhraní můžeme konfigurovat požadované služby. Jako tvůrci tak můžeme nabízet své mobilní aplikace v Google Play.

Zpřístupnění Android aplikace v obchodě Google Play se skládá z následujících dvou hlavních kroků.

4.6.2 Příprava aplikace k publikaci

Kontrola, zda se nevyskytují chyby při běhu aplikace, které by způsobily okamžité ukončení programu. Uživatel by byl dotázán, zda chce chybu nahlásit vývojáři, či nikoli. Nahlášená chyba by byla k dispozici v administraci Google Play.

Ověření funkčnosti podepsané aplikace je vždy nezbytné. Pokud funguje pracovní verze správně, není samozřejmé, že bude vše v pořádku i u sestavení pro konečnou distribuci.

Aktualizace zdrojů v aplikaci se provádí, aby výsledný balíček APK neobsahoval zbytečné testovací soubory a nezvětšoval tak výslednou velikost aplikace pro distribuci. Odstraníme tedy testovací obrázky a podpůrné soubory.

Kontrola správného spárování podpůrné serverové aplikace je nezbytná k budoucím samostatným aktualizacím dat v koncových zařízeních uživatelů. Pokud by podpůrná služba nebyla v provozu, Android aplikace by sice spolehlivě fungovala i nadále, ale nezískávala by průběžné aktualizace dat.

4.6.3 Uvolnění aplikace pro uživatele

Uvolnění otestované aplikace na Google Play je jednoduchý proces skládající se z několika následujících kroků.

Příprava propagačních textů, které popisují základní vlastnosti aplikace.

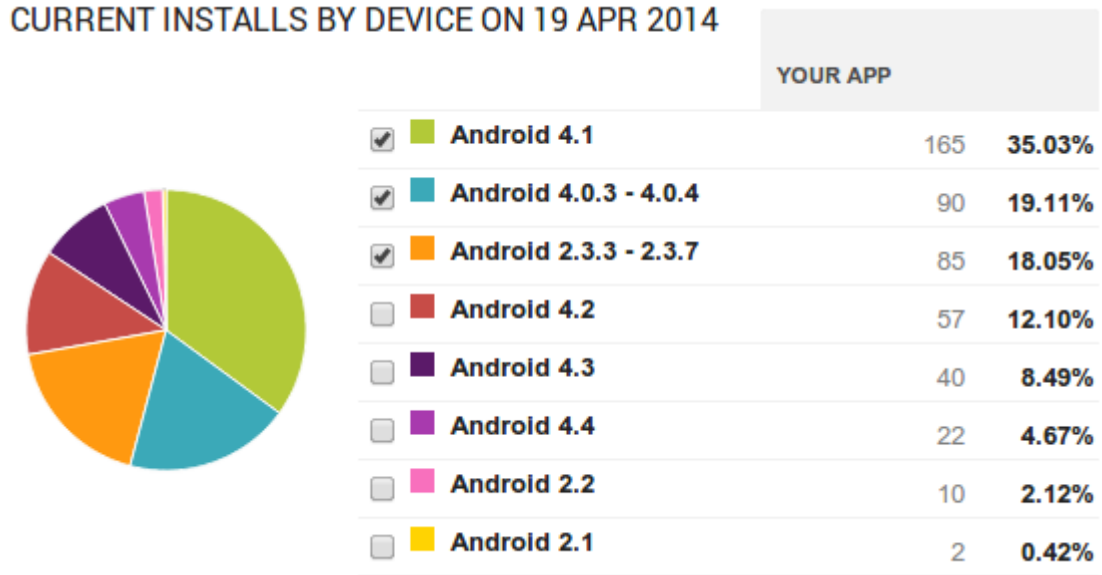
Dostupnost je možno omezit pro konkrétní zařízení, případně zemi. Pro naše potřeby je nastaveno, že aplikace bude ke stažení jen v České republice a na Slovensku.

Pokud máme vše správně nakonfigurováno, jsme připraveni nabídnout náš výsledný produkt veřejnosti. Nová verze bude v Google Play dostupná přibližně za hodinu od kliknutí na tlačítko publikace této nové verze aplikace.

Tento vydávací cyklus se opakuje s každou nově uvolňovanou verzí Android aplikace.

Na obrázku vidíme aktuální zastoupení instalací Android aplikace, rozdělené podle verze systému Android.

Obrázek 4.13 Google Play - instalace Android aplikace rozvrhu



Zdroj: Administrace vývojářského účtu Google Play

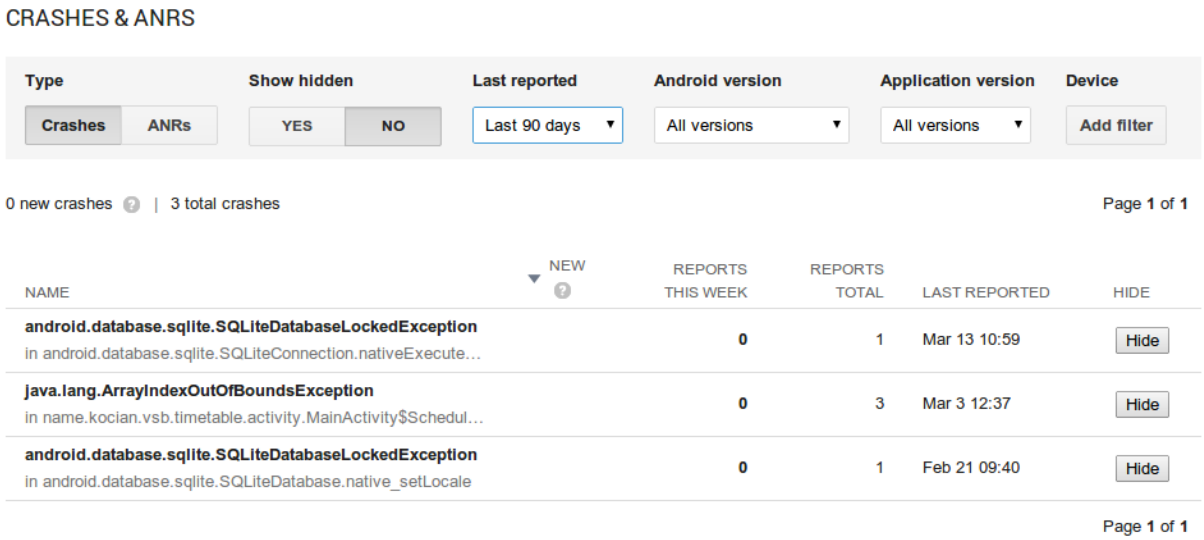
4.7 Hlášení chyb při běhu aplikace

Uživatelé mohou aplikaci neomezeně využívat, mít ji nainstalovanou na stovkách různých zařízení, a tak se může stát, že se někde vyskytne nějaký problém. K tomu abychom měli přehled o spolehlivosti, nám může posloužit hlášení o pádu aplikace. Nahlášení chyby se však odvíjí od ochoty uživatele daný report poslat.

Na obrázku můžeme vidět některá hlášení o pádu aplikace za posledních 90 dní. Čím více těchto hlášení dostaneme, tím naléhavěji bychom měli daný problém řešit, protože

tito uživatelé nejspíš nemohou aplikaci plnohodnotně využívat a hrozí, že si ji brzy odinstalují.

Obrázek 4.14 Google Play - hlášení o pádu aplikace



Zdroj: Vlastní zpracování

Pokud uživatel chybový report odešle, získáme tím informaci o tom na jakém místě v kódu se vyskytla chyba. To nám může velmi pomoci při řešení problémů.

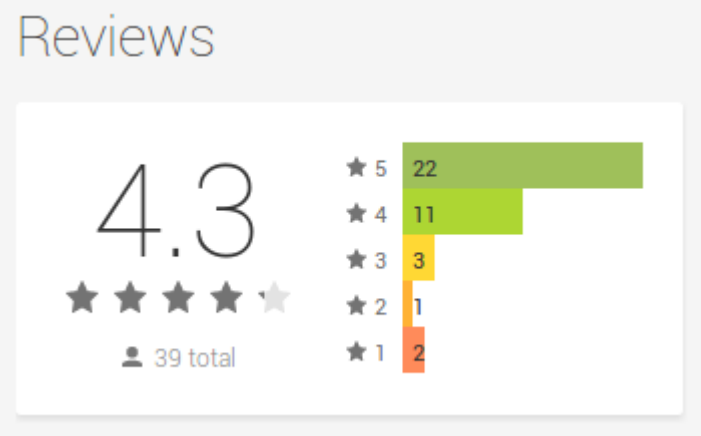
Obrázek 4.15 Konkrétní chybové hlášení

Zdroj: Vlastní zpracování

4.8 Hodnocení aplikace

Aplikaci rozvrhu je možno ocenit hvězdičkami, případně přidat slovní ohodnocení. První verze byly zveřejněny na začátku školního roku 2013/2014 a od té doby aplikace získala ohodnocení 4.3 bodu, momentálně je aplikace nainstalovaná na přibližně 500 zařízeních.

Obrázek 4.16 Google Play - hodnocení aplikace



Zdroj: Vlastní zpracování

5 Závěr

Mobilní aplikace mají obrovský potenciál a jejich vývoj jde velmi rychle dopředu. Firmám se vyplatí investovat to těchto moderních technologií a mít tak náskok před konkurencí. To se týká i školního prostředí, kde tyto podpůrné prostředky zlepšují poskytované služby.

Na Vysoké škole báňské - Technické univerzitě Ostrava žádná mobilní aplikace zprostředkující školní rozvrhy dosud neexistovala, a tak zde byla příležitost k získání nových dovedností s platformou Android. Hnacím motorem byla především myšlenka na to, že když se prvotní návrh setkal s poměrně kladným přijetím, mohla by si aplikace najít velké množství uživatelů, kteří by rádi měli při ruce informace o školních rozvrzích. Tato zpočátku zájmová aktivita, se následně rozvinula do této diplomové práce.

Pro potřeby této aplikace bylo nutné získat rozvrhová data, což se úspěšně podařilo s pomocí zaměstnanců školy. Tyto data byla následně vyhodnocena a zpracována do podoby, ve které je mohu efektivně využívat mobilní aplikace.

Byl zpracován návrh Android aplikace, která studentům a učitelům umožní zobrazovat vyučované předměty podle několika kritérií. Zobrazovat je možno rozvrhy podle studijních skupin, místností, předmětů a vyučujících.

Vytvořené podpůrné webové služby zajišťující data pro uživatelskou aplikaci. Tyto služby jsou neustále k dispozici. Do budoucna by mohly být využity pro implementaci na jiných mobilních platformách jako jsou například Apple iOS, nebo Windows Mobile.

Hlavním produktem je zde samotná Android aplikace, kterou v současné době má nainstalováno přibližně 500 uživatelů. Mohlo by to být více, ale tohoto stavu se dosáhlo téměř bez aktivní propagace a s prakticky nulovými náklady. Do budoucna by bylo vhodné rozšířit povědomí o její existenci.

Stále je zde prostor pro zlepšení. Rozvrhy pro kombinované studium ještě nejsou zpracovány do podoby, v jaké bych si je představoval. K lepší spolehlivosti a bezchybnosti, by bylo dobré doplnit další jednotkové testy a mít tak kód pod větší kontrolou. Mohla by se doplnit optimalizace pro tablety, kde je více prostoru a tak se zde můžou zobrazovat podrobnější informace než na telefonech. Aktuálně největší slabinou je pomalé načítání dat z interní databáze SQLite, které zabere na pomalejších zařízeních i několik sekund a snižuje

tak uživatelský požitek. Operační systém Android se velmi rychle vyvíjí a je třeba postupem času držet krok, ověřovat správný chod a reagovat na nové požadavky uživatelů. Nicméně vývoj softwarových aplikací je nikdy nekončící běh na dlouhou trať, protože jakmile dokončíme požadovanou funkčnost, napadají nás nové možnosti, a je tak nutné rozlišit jestli by jejich implementace byla prospěšná.

Při zpracování praktické části jsme si osvojili mnoho zajímavých technologií a získali zkušenosti s kompletním vývojovým cyklem Android aplikace. I když se někdy vyskytla jistá úskalí, vždy se podařilo všechny problémy vyřešit a dovést tak práci do úspěšného konce. Výsledný produkt má svůj potenciál za předpokladu, že se o něj bude někdo průběžně starat, jinak postupem času upadne v zapomnění.

Seznam použité literatury

FOWLER, Martin. *UML distilled: a brief guide to the standard object modeling language.* 3rd ed. Boston: Addison-Wesley, 2004. ISBN 03-211-9368-7.

MEIER, Reto. *Professional Android 4 application development.* 2rd ed., revised. Indianapolis: John Wiley, 2012. ISBN 978-111-8262-153.

PECINOVSKÝ, Rudolf. Myslíme objektově v jazyku Java 5.0: oficiální průvodce tvorbou, správou a laděním databází. Vyd. 1. Praha: Grada, 2004, 601 s. ISBN 80-247-0941-4.

SCHNEIDER, Robert D. MySQL: oficiální průvodce tvorbou, správou a laděním databází. Praha: Grada Publishing, 2006, 372 s. ISBN 80-247-1516-3.

SCHWALBE, Kathy. *Řízení projektů v IT: kompletní průvodce.* Vyd. 1. Brno: Computer Press, 2011. ISBN 978-80-251-2882-4.

Internetové prameny

Git: Fast version control. [online]. [cit. 2014-04-02]. Dostupné z: http://git-scm.com/

IDC: Android and iOS Continue to Dominate the Worldwide Smartphone Market [online]. [cit. 2014-03-15]. Dostupné z: http://www.idc.com/getdoc.jsp?containerId=prUS24676414

Tech2crack: History of Android [online]. [cit. 2014-04-25]. Dostupné z: http://www.tech2crack.com/history-android/

GOOGLE INC. Android Open Source Project [online]. [cit. 2014-04-25]. Dostupné z: https://source.android.com/

GOOGLE INC. Android developers [online]. [cit. 2014-03-05]. Dostupné z: https://developer.android.com

GOOGLE INC. Google Play Developer Console [online]. [cit. 2014-04-25]. Dostupné z: https://play.google.com/apps/publish/

MYSLIVEČEK, David. Svět Androida: Krátké ohlédnutí za historií Androidu [online]. [cit. 2014-04-25]. Dostupné z: http://www.svetandroida.cz/kratke-ohlednuti-za-historii-androidu-201305

Seznam použitých zkratek

DOS útok – Denial of service, nedostupnost služby díky přehlcení požadavky

HTML5 – HyperText Markup Language, je značkovací jazyk používaný především k tvorbě webových stránek

Java – programovací jazyk

JSON – JavaScript Object Notation, textový formát dat

Malware – nevhodně se chovající software

MySQL – relační databáze SQL

SDK – Software development kit, softwarové nástroje pro vytváření aplikací

SQL – Structured Query Language, jazyk pro získávání dat z databáze

SQLite – kompaktní databáze využívající jazyk SQL

XML – Extensible Markup Language, značkovací jazyk

Prohlášení o využití výsledků diplomové práce

Prohlašuji, že

- jsem byl(a) seznámen(a) s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb. – autorský zákon, zejména § 35 – užití díla v rámci občanských a náboženských obřadů, v rámci školních představení a užití díla školního a § 60 – školní dílo;
- beru na vědomí, že Vysoká škola báňská – Technická univerzita Ostrava (dále jen VŠB-TUO) má právo nevýdělečně, ke své vnitřní potřebě, diplomovou práci užít (§ 35 odst.3);
- souhlasím s tím, že diplomová práce bude v elektronické podobě archivována v Ústřední knihovně VŠB-TUO a jeden výtisk bude uložen u vedoucího diplomové práce. Souhlasím s tím, že bibliografické údaje o diplomové práci budou zveřejněny v informačním systému VŠB-TUO;
- bylo sjednáno, že s VŠB-TUO, v případě zájmu z její strany, uzavřu licenční smlouvu s oprávněním užít dílo v rozsahu § 12 odst. 4 autorského zákona;
- bylo sjednáno, že užít své dílo, diplomovou práci, nebo poskytnout licenci k jejímu využití mohu jen se souhlasem VŠB-TUO, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly VŠB-TUO na vytvoření díla vynaloženy (až do jejich skutečné výše).

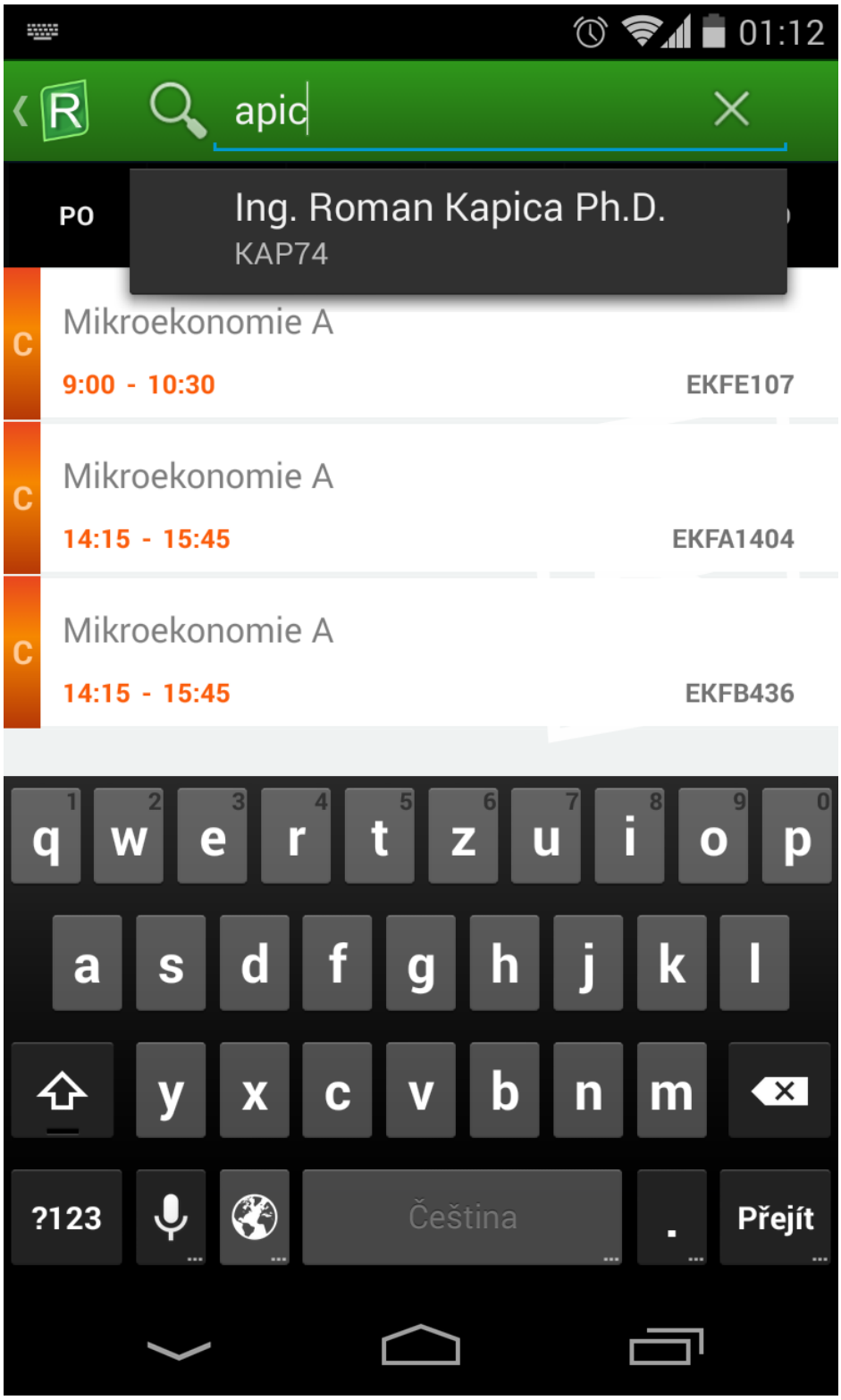
V Ostravě dne 25.4.2014

.....

Bc. Michal Kocián

Seznam příloh

Příloha č. 1 Náhled Android aplikace



Příloha č. 2 Webová stránka Google Play